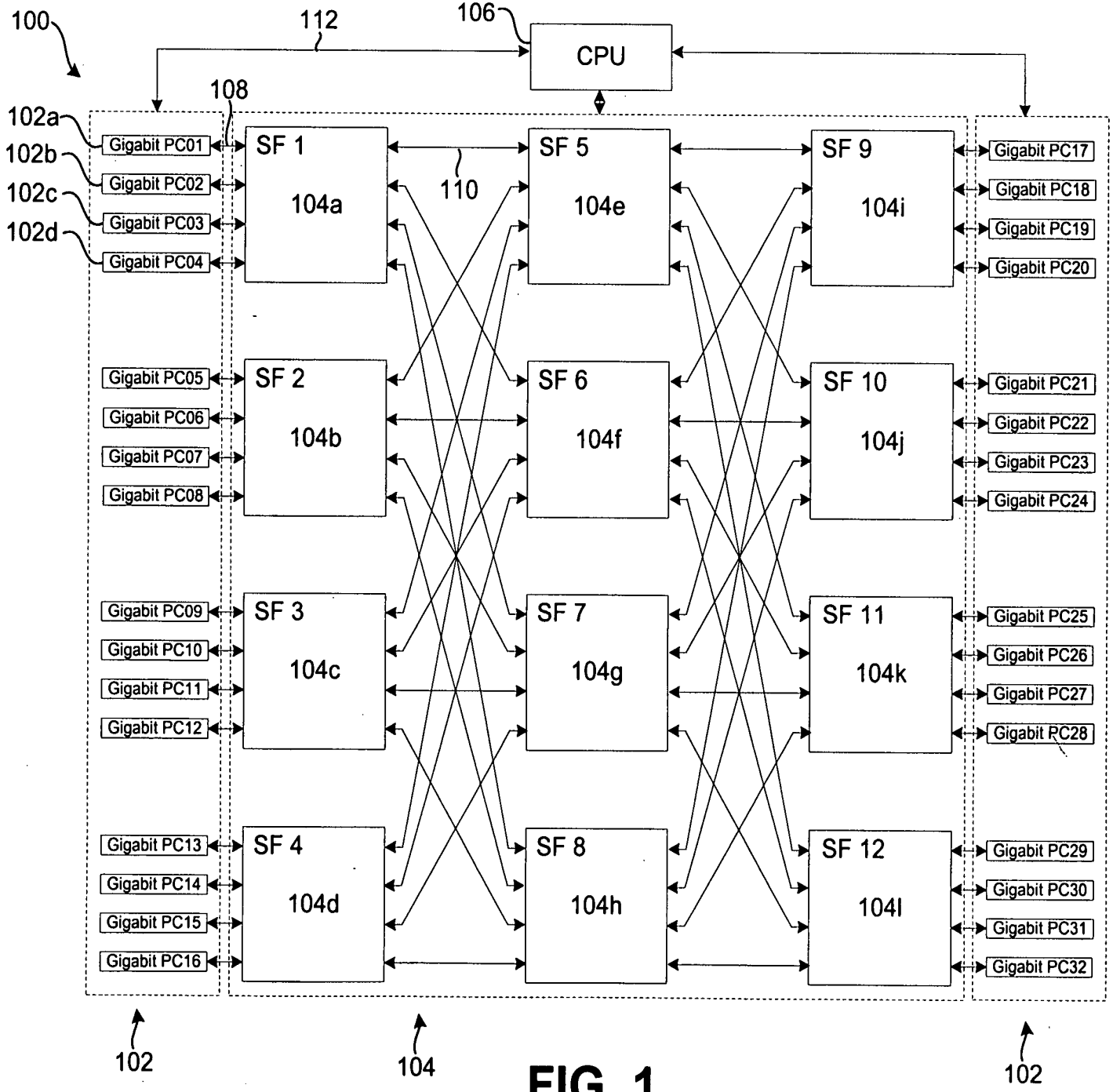


FIG. 1

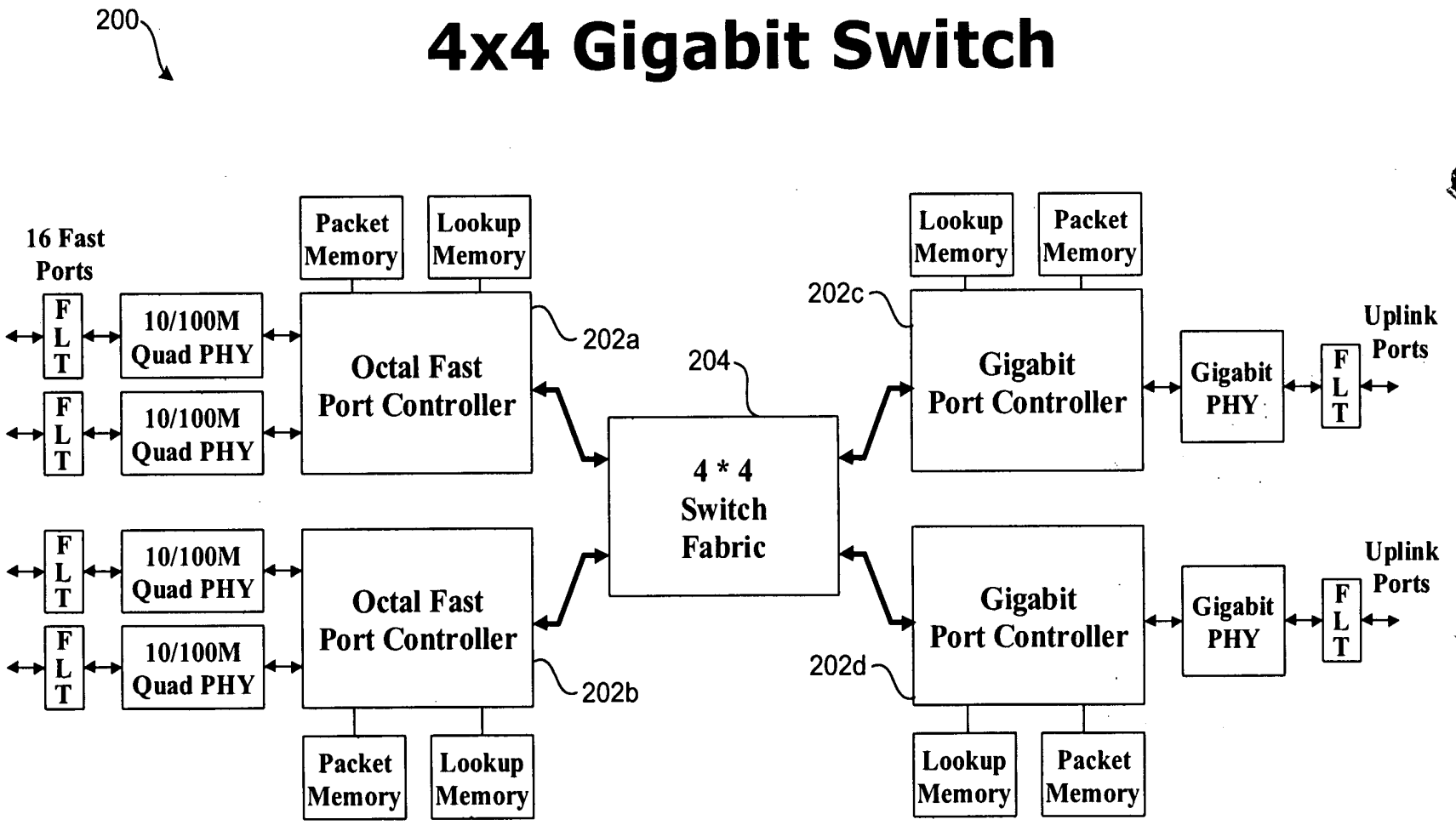

200
4x4 Gigabit Switch
16 Fast Ports
F L T
10/100M Quad PHY
Packet Memory
Lookup Memory
Octal Fast Port Controller
202a
204
4 * 4 Switch Fabric
202b
202c
202d
Gigabit Port Controller
Gigabit PHY
Uplink Ports

FIG. 2

❑ Gigabit Up-linking

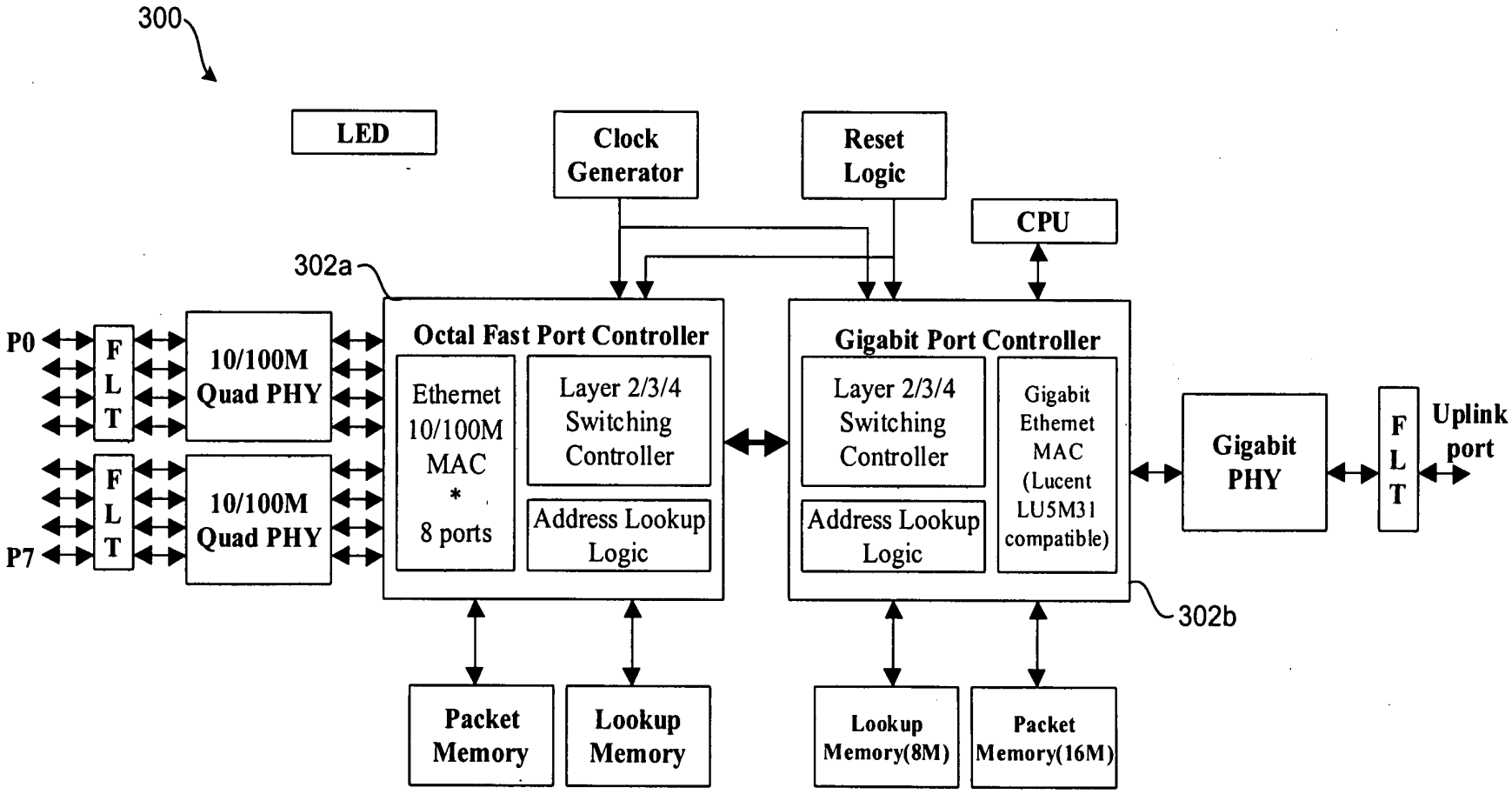


FIG. 3

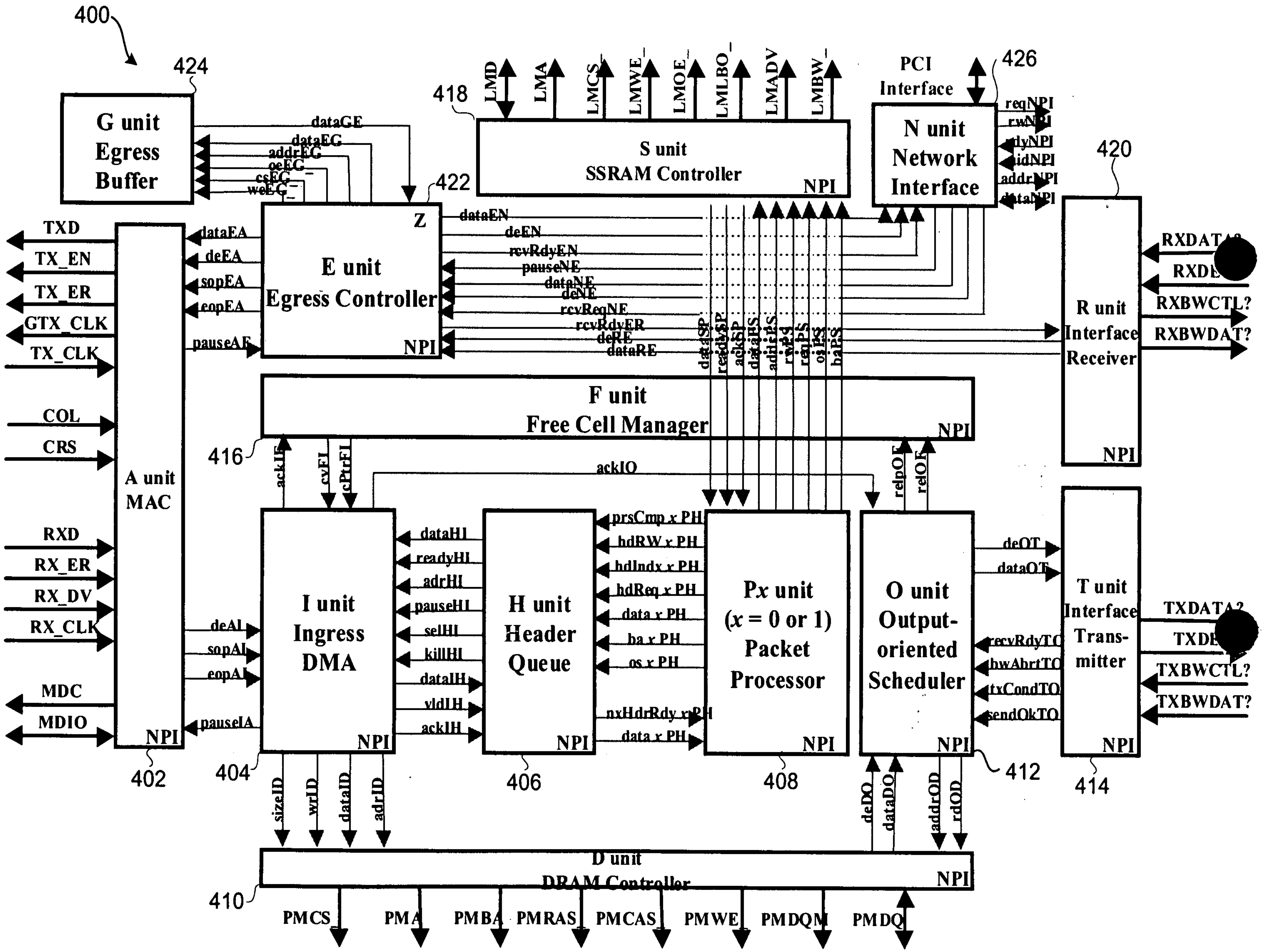

400
424
G unit
Egress
Buffer
dataGE
dataEG
addrEG
oeEG_
csEG_
weEG_
422
Z
E unit
Egress Controller
NPI
418
LMD
LMA
LMCS_
LMWE_
LMOE_
LMLBO_
LMADV
LMBW_
S unit
SSRAM Controller
NPI
PCI
Interface
426
N unit
Network
Interface
reqNPI
rwNPI
rdyNPI
idNPI
addrNPI
dataNPI
420
R unit
Interface
Receiver
NPI
RXDATA
RXDE
RXBWCTL?
RXBWDAT?
TXD
TX_EN
TX_ER
GTX_CLK
TX_CLK
COL
CRS
RXD
RX_ER
RX_DV
RX_CLK
MDC
MDIO
A unit
MAC
NPI
402
dataEA
deEA
sopEA
eopEA
pauseAE
dataEN
deEN
rcvRdyEN
pauseNE
dataNE
deNE
rcvReqNE
rcvRdyER
deRE
dataRE
dataSP
readySP
ackSP
dataPS
addrPS
rwPS
reqPS
osPS
baPS
F unit
Free Cell Manager
NPI
416
ackIF
cvFI
cPtrFI
ackIO
relpOF
relOF
I unit
Ingress
DMA
NPI
404
deAI
sopAI
eopAI
pauseIA
dataHI
readyHI
adrHI
pauseHI
selHI
killHI
dataIH
vldIH
ackIH
H unit
Header
Queue
NPI
406
prsCmp x PH
hdRW x PH
hdIndx x PH
hdReq x PH
data x PH
ba x PH
os x PH
nxHdrRdy x PH
data x PH
Px unit
(x = 0 or 1)
Packet
Processor
NPI
408
O unit
Output-
oriented
Scheduler
NPI
412
deOT
dataOT
rcvRdyTO
hwAbrtTO
txCondTO
sendOkTO
T unit
Interface
Trans-
mitter
NPI
414
TXDATA
TXDE
TXBWCTL?
TXBWDAT?
sizeID
wrID
dataID
adrID
deDO
dataDO
addrOD
rdOD
D unit
DRAM Controller
NPI
410
PMCS
PMA
PMBA
PMRAS
PMCAS
PMWE
PMDQM
PMDQ

FIG. 4

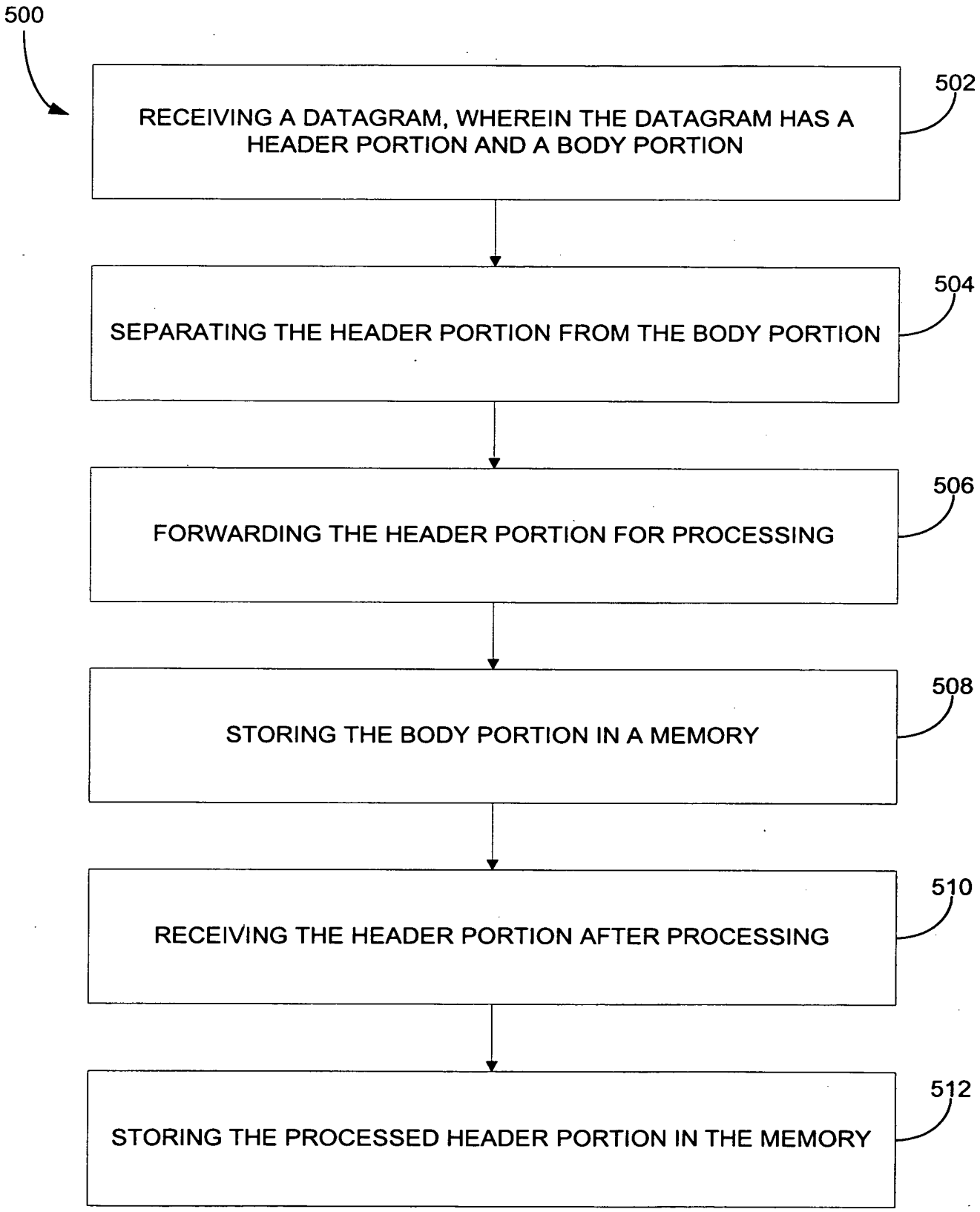

500
RECEIVING A DATAGRAM, WHEREIN THE DATAGRAM HAS A HEADER PORTION AND A BODY PORTION
502
SEPARATING THE HEADER PORTION FROM THE BODY PORTION
504
FORWARDING THE HEADER PORTION FOR PROCESSING
506
STORING THE BODY PORTION IN A MEMORY
508
RECEIVING THE HEADER PORTION AFTER PROCESSING
510
STORING THE PROCESSED HEADER PORTION IN THE MEMORY
512

FIG. 5

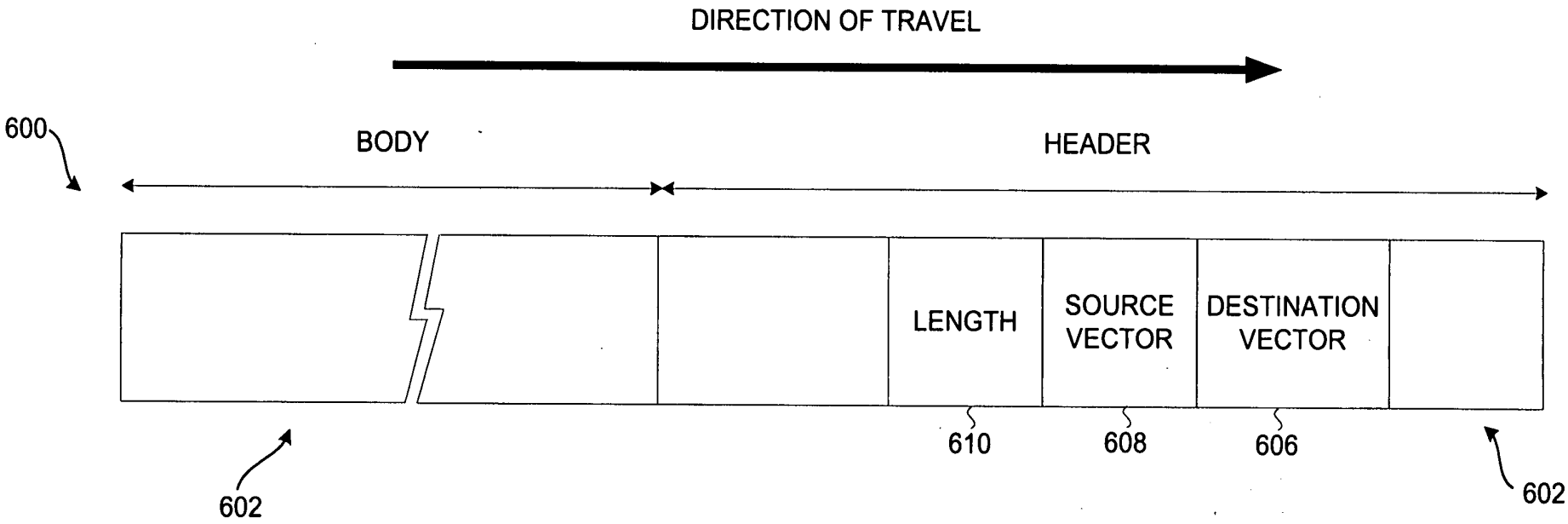


FIG. 6

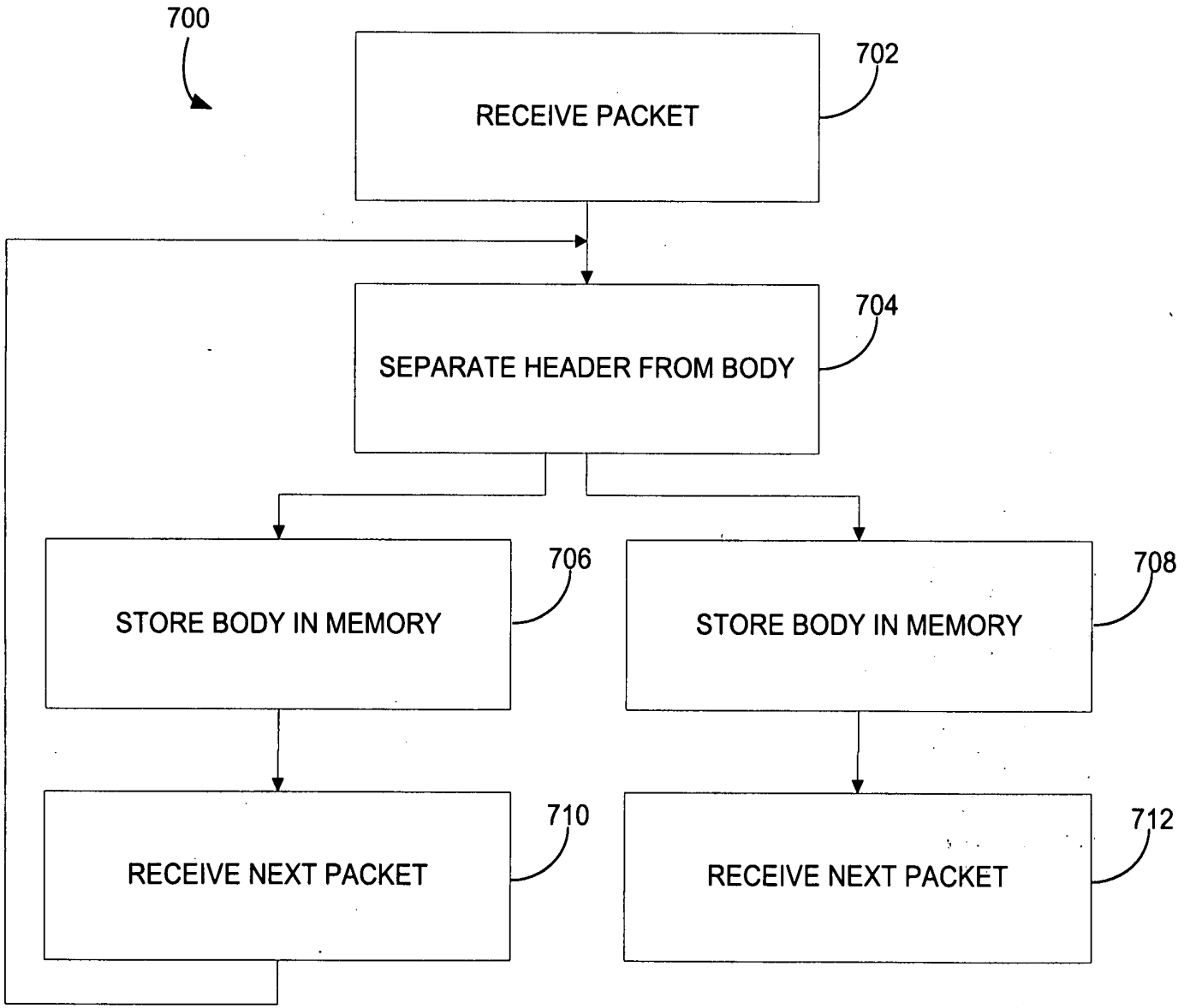

700
RECEIVE PACKET
702
SEPARATE HEADER FROM BODY
704
STORE BODY IN MEMORY
706
STORE BODY IN MEMORY
708
RECEIVE NEXT PACKET
710
RECEIVE NEXT PACKET
712

FIG. 7

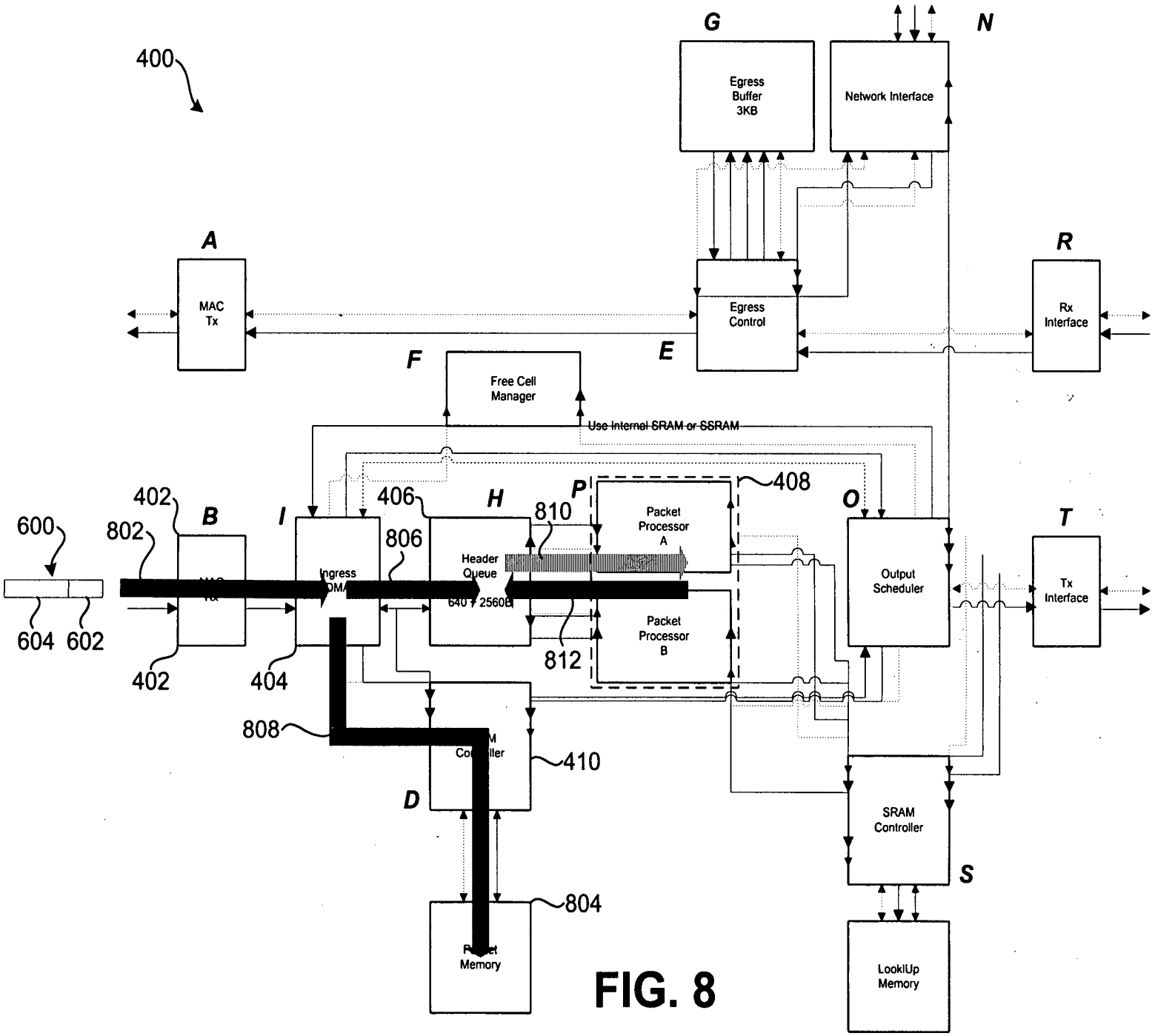


FIG. 8

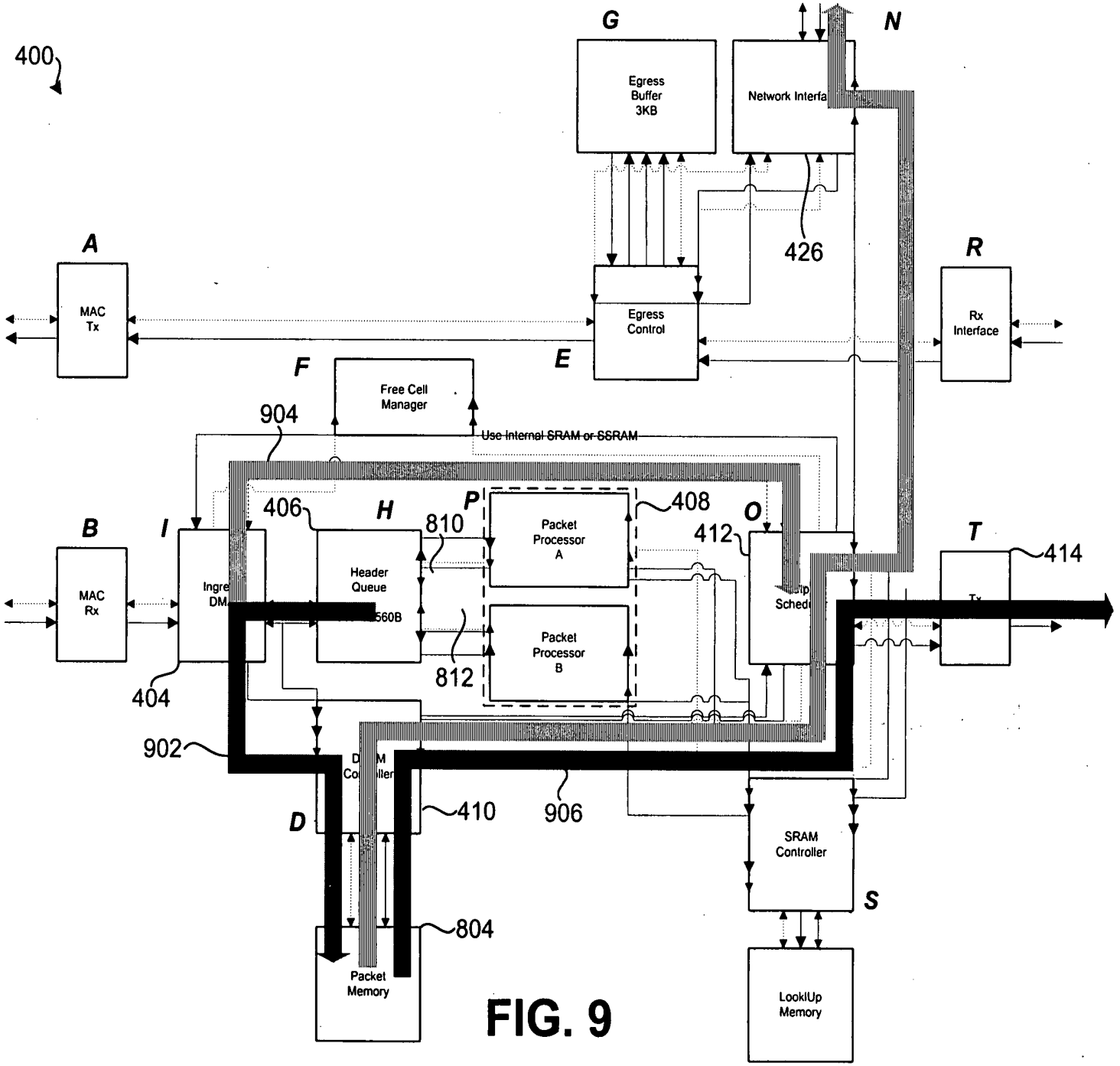

400
G
N
Egress Buffer 3KB
Network Interfa
426
A
MAC Tx
Egress Control
R
Rx Interface
E
F
Free Cell Manager
904
Use Internal SRAM or SSRAM
406
H
810
P
408
O
412
T
414
B
I
MAC Rx
Ingre DM
Header Queue
560B
Packet Processor A
Packet Processor B
Sched
404
812
902
D
410
906
SRAM Controller
S
804
Packet Memory
LookUp Memory

FIG. 9

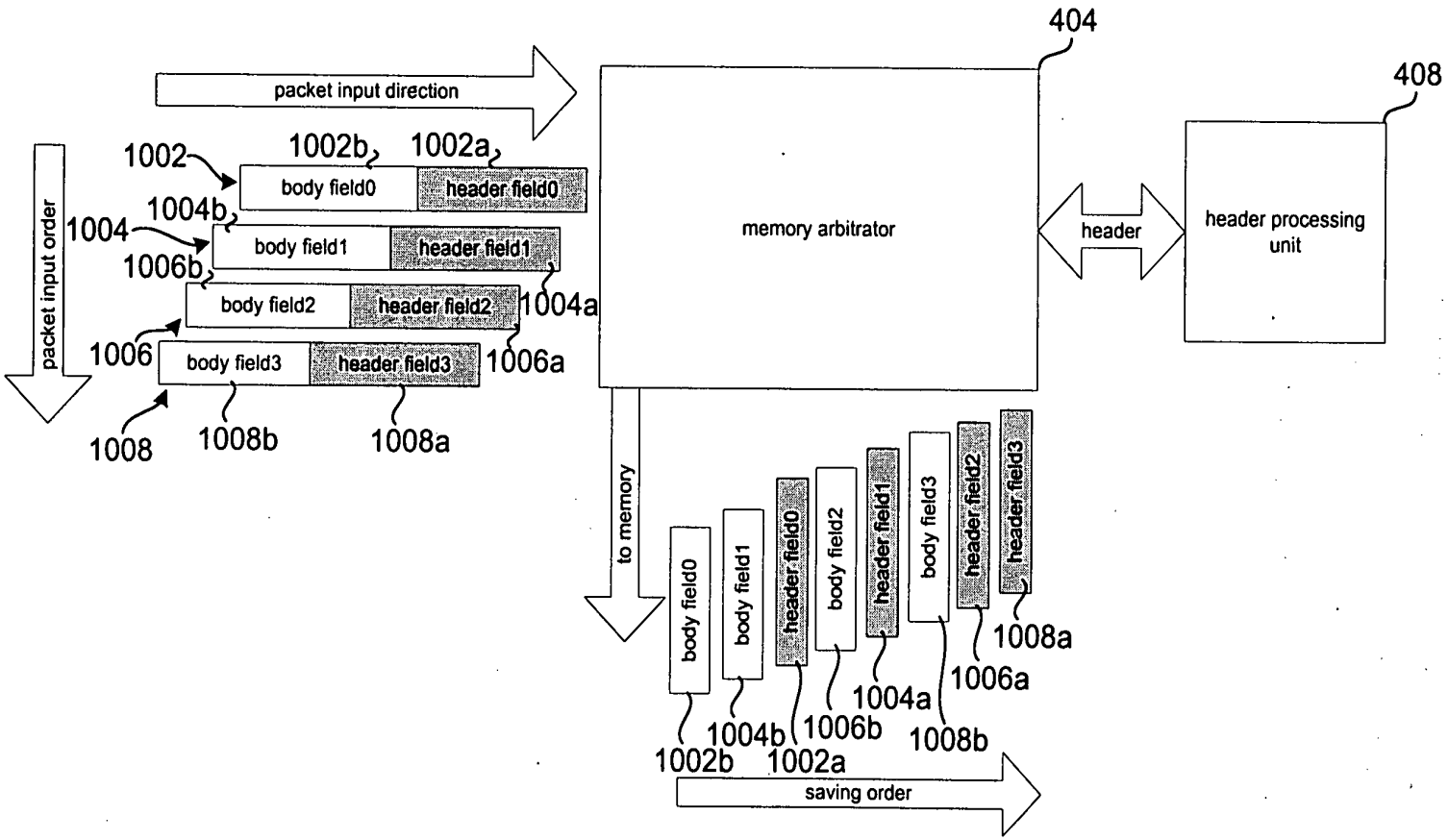

404
408
packet input direction
packet input order
1002
1002b
1002a
body field0
header field0
1004
1004b
body field1
header field1
1006b
body field2
header field2
1004a
1006
body field3
header field3
1006a
1008
1008b
1008a
memory arbitrator
header
header processing unit
to memory
body field0
body field1
header field0
body field2
header field1
body field3
header field2
header field3
1008a
1006a
1004a
1004b
1006b
1008b
1002b
1002a
saving order

FIG. 10

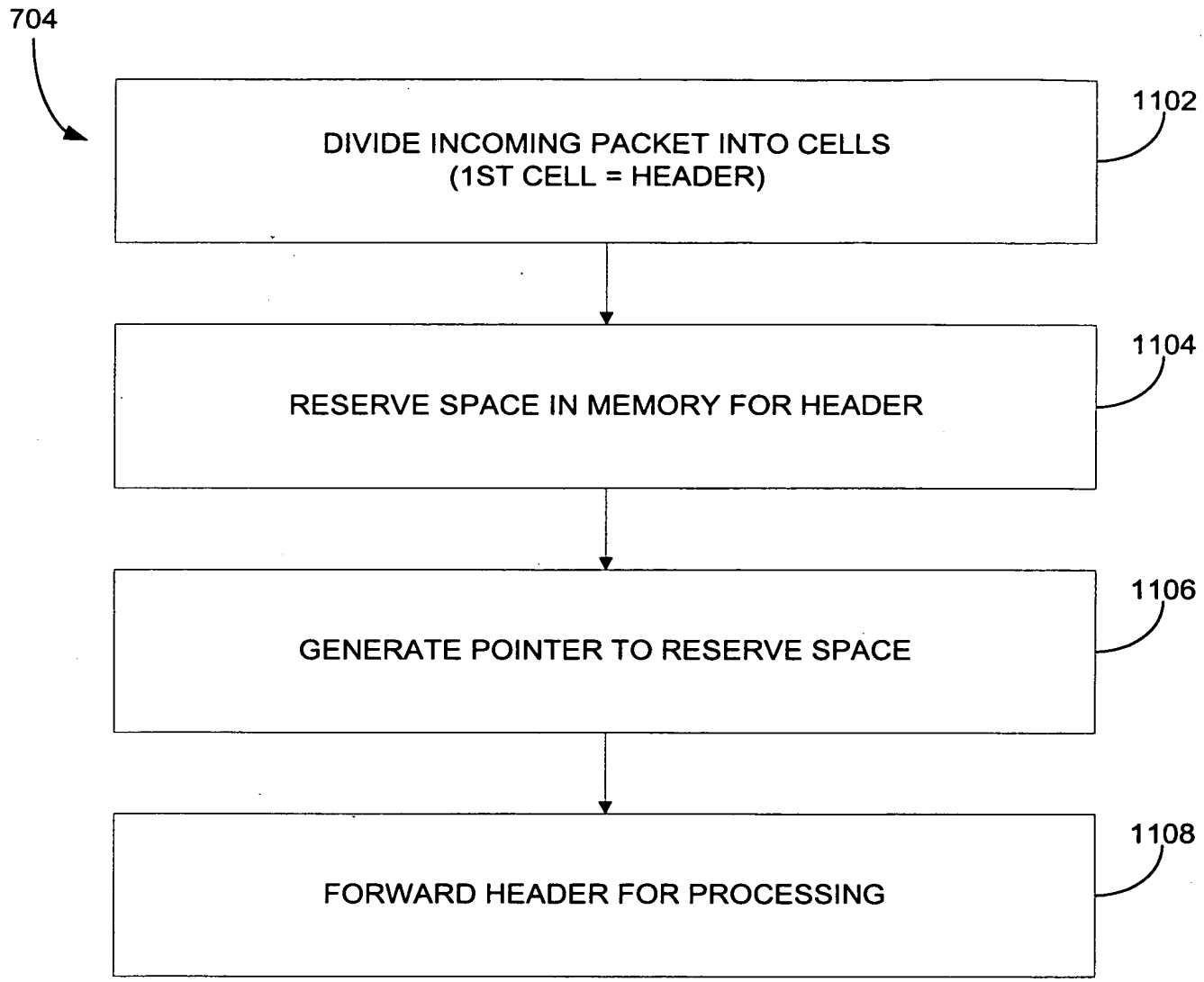

704
DIVIDE INCOMING PACKET INTO CELLS
(1ST CELL = HEADER)
1102
RESERVE SPACE IN MEMORY FOR HEADER
1104
GENERATE POINTER TO RESERVE SPACE
1106
FORWARD HEADER FOR PROCESSING
1108

FIG. 11

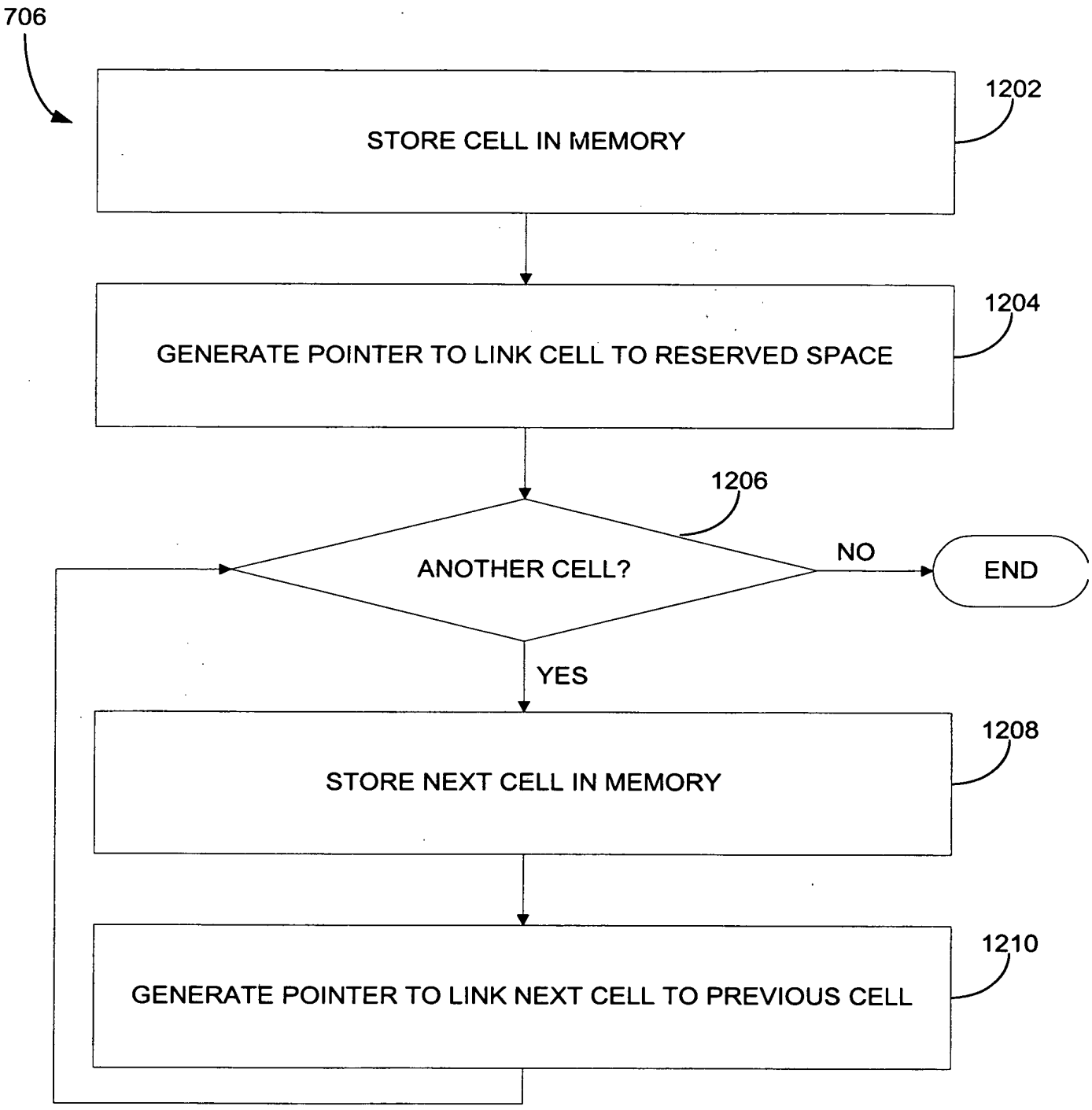

706
STORE CELL IN MEMORY
1202
GENERATE POINTER TO LINK CELL TO RESERVED SPACE
1204
1206
ANOTHER CELL?
NO
END
YES
STORE NEXT CELL IN MEMORY
1208
GENERATE POINTER TO LINK NEXT CELL TO PREVIOUS CELL
1210

FIG. 12

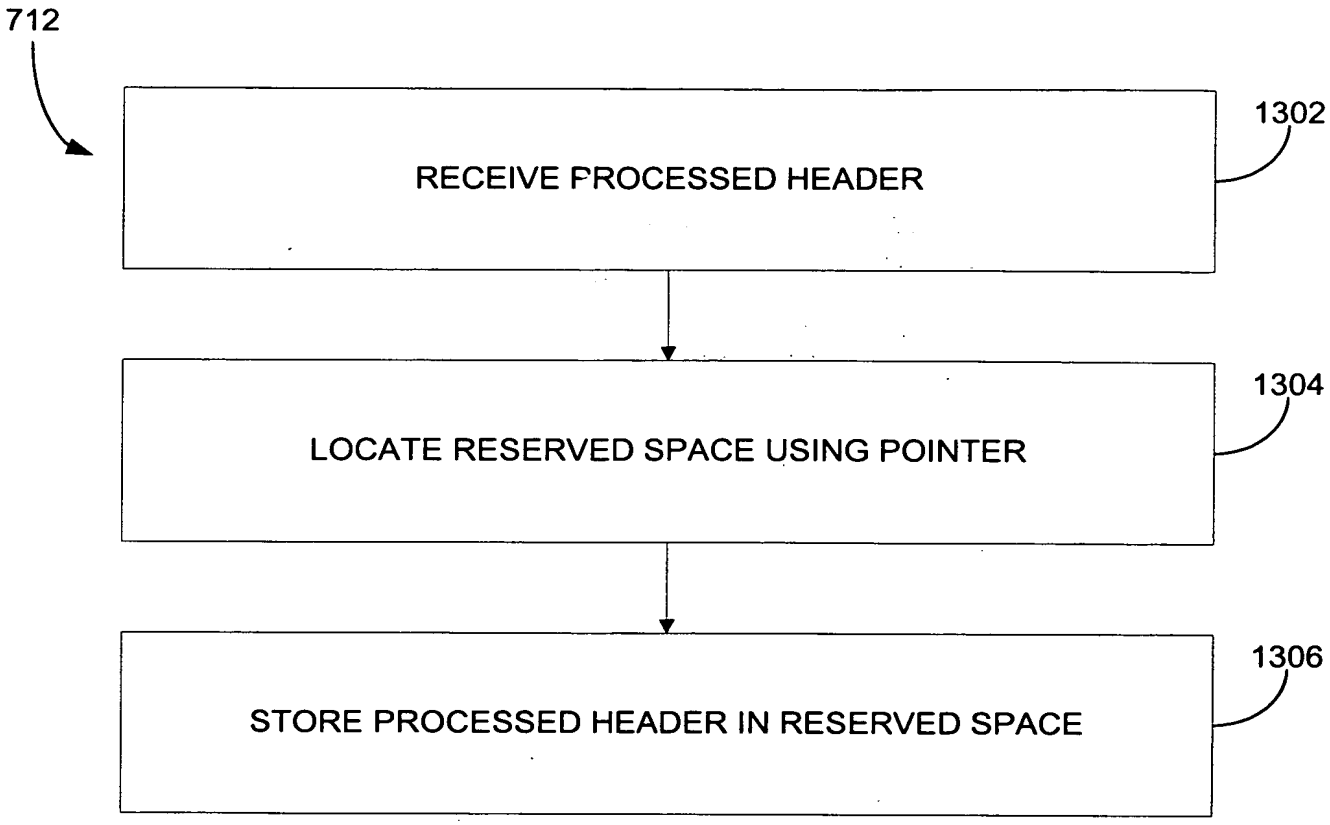
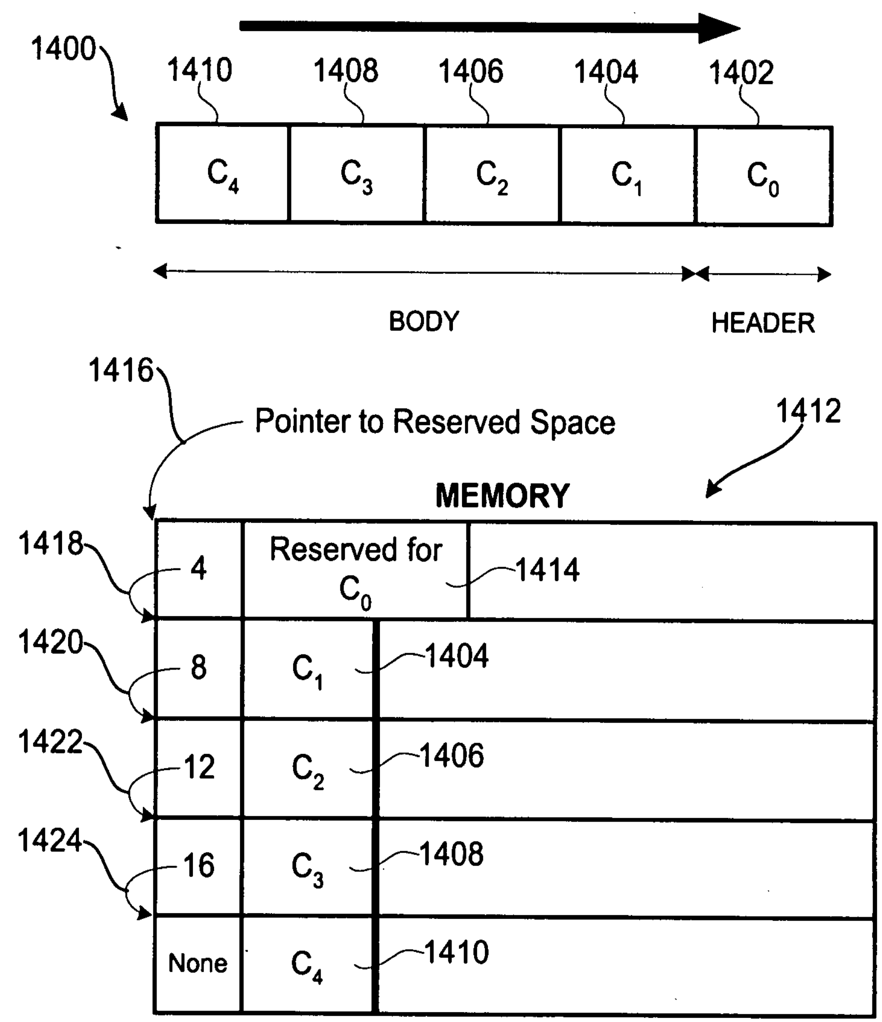

712
RECEIVE PROCESSED HEADER
1302
LOCATE RESERVED SPACE USING POINTER
1304
STORE PROCESSED HEADER IN RESERVED SPACE
1306

FIG. 13



1400
1410
1408
1406
1404
1402
C_4
C_3
C_2
C_1
C_0
BODY
HEADER
1416
Pointer to Reserved Space
1412
MEMORY
1418
1420
1422
1424
4
8
12
16
None
Reserved for C_0
1414
C_1
1404
C_2
1406
C_3
1408
C_4
1410

FIG. 14

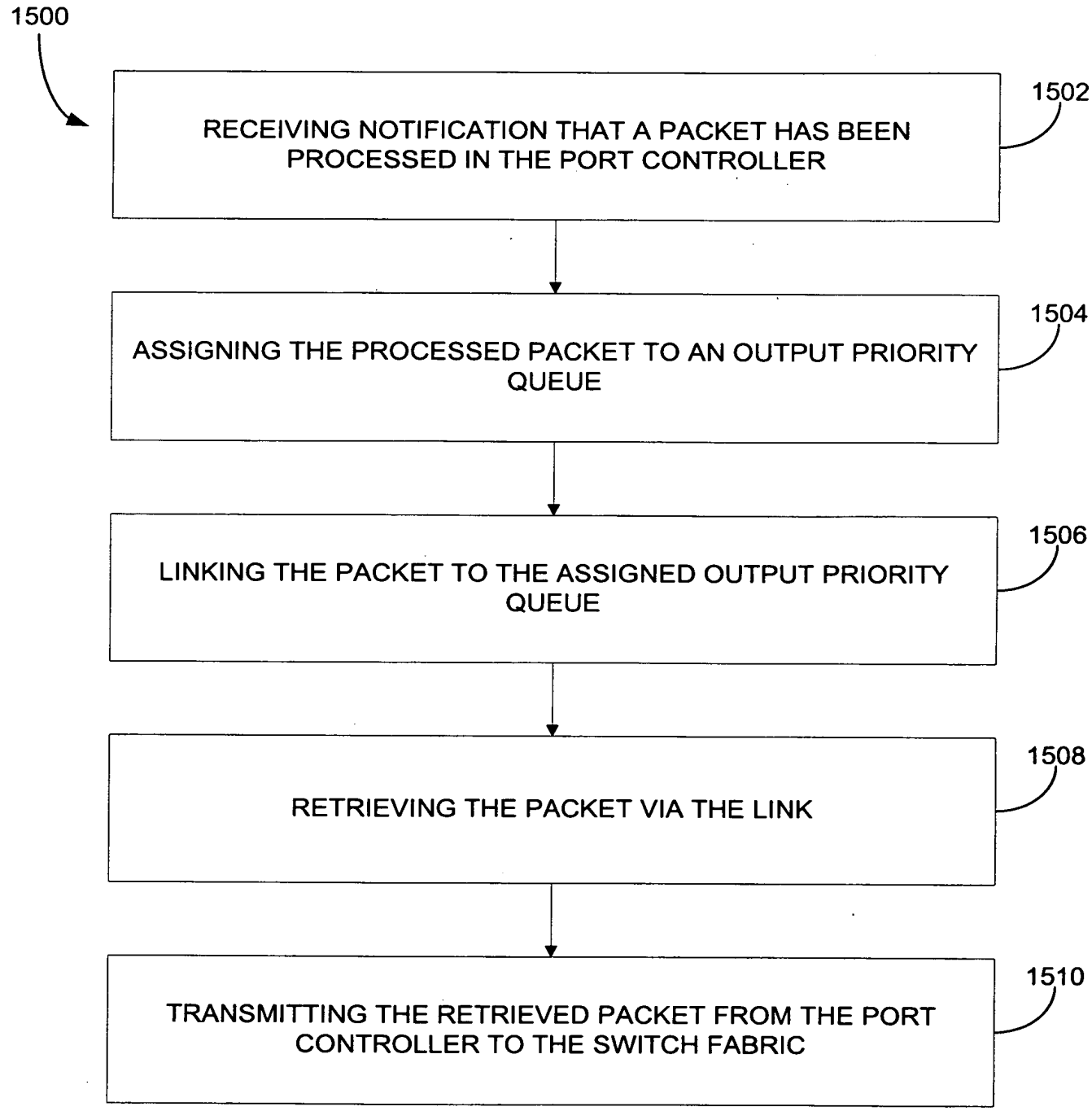

1500
RECEIVING NOTIFICATION THAT A PACKET HAS BEEN PROCESSED IN THE PORT CONTROLLER
1502
ASSIGNING THE PROCESSED PACKET TO AN OUTPUT PRIORITY QUEUE
1504
LINKING THE PACKET TO THE ASSIGNED OUTPUT PRIORITY QUEUE
1506
RETRIEVING THE PACKET VIA THE LINK
1508
TRANSMITTING THE RETRIEVED PACKET FROM THE PORT CONTROLLER TO THE SWITCH FABRIC
1510

FIG. 15

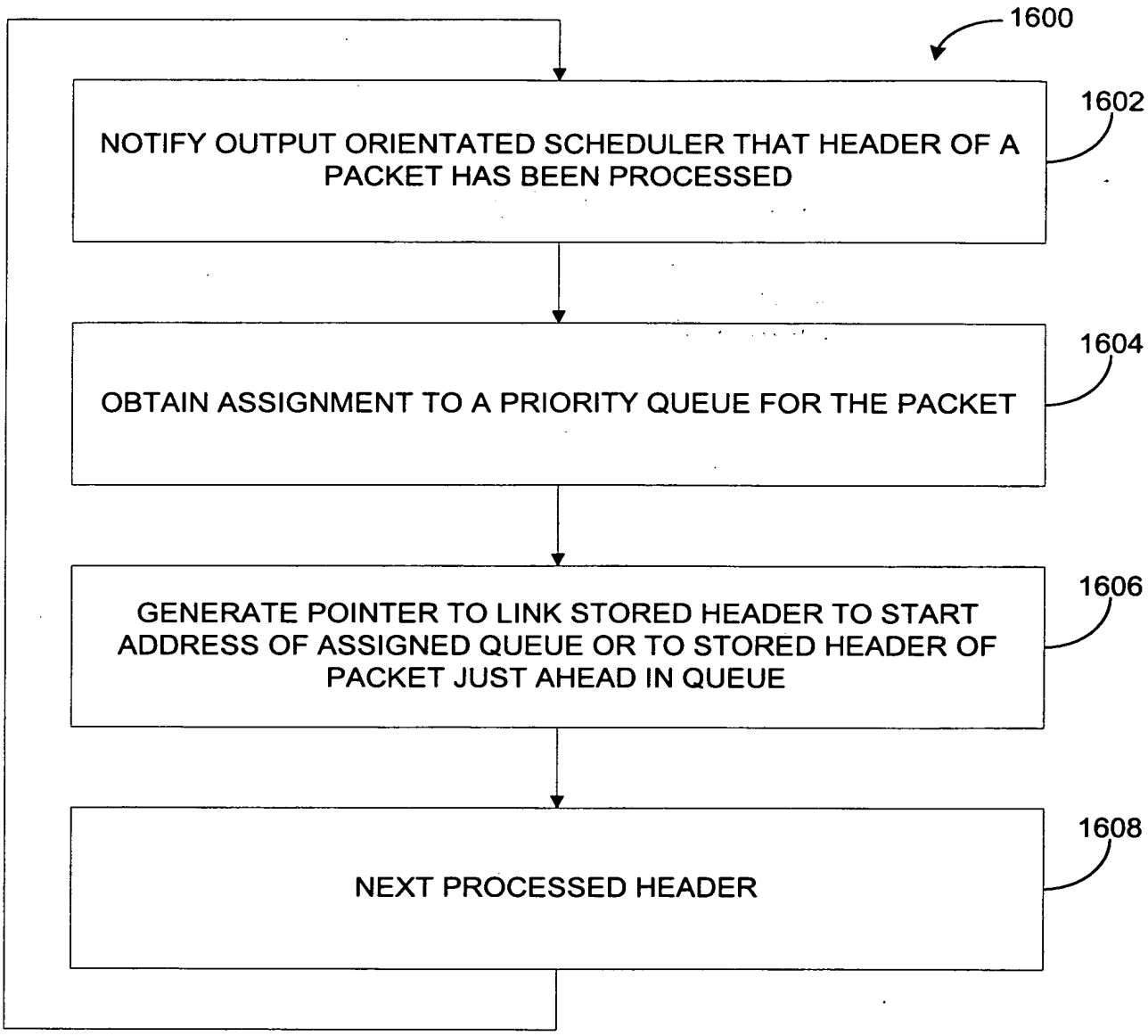

1600
NOTIFY OUTPUT ORIENTATED SCHEDULER THAT HEADER OF A PACKET HAS BEEN PROCESSED
1602
OBTAIN ASSIGNMENT TO A PRIORITY QUEUE FOR THE PACKET
1604
GENERATE POINTER TO LINK STORED HEADER TO START ADDRESS OF ASSIGNED QUEUE OR TO STORED HEADER OF PACKET JUST AHEAD IN QUEUE
1606
NEXT PROCESSED HEADER
1608

FIG. 16

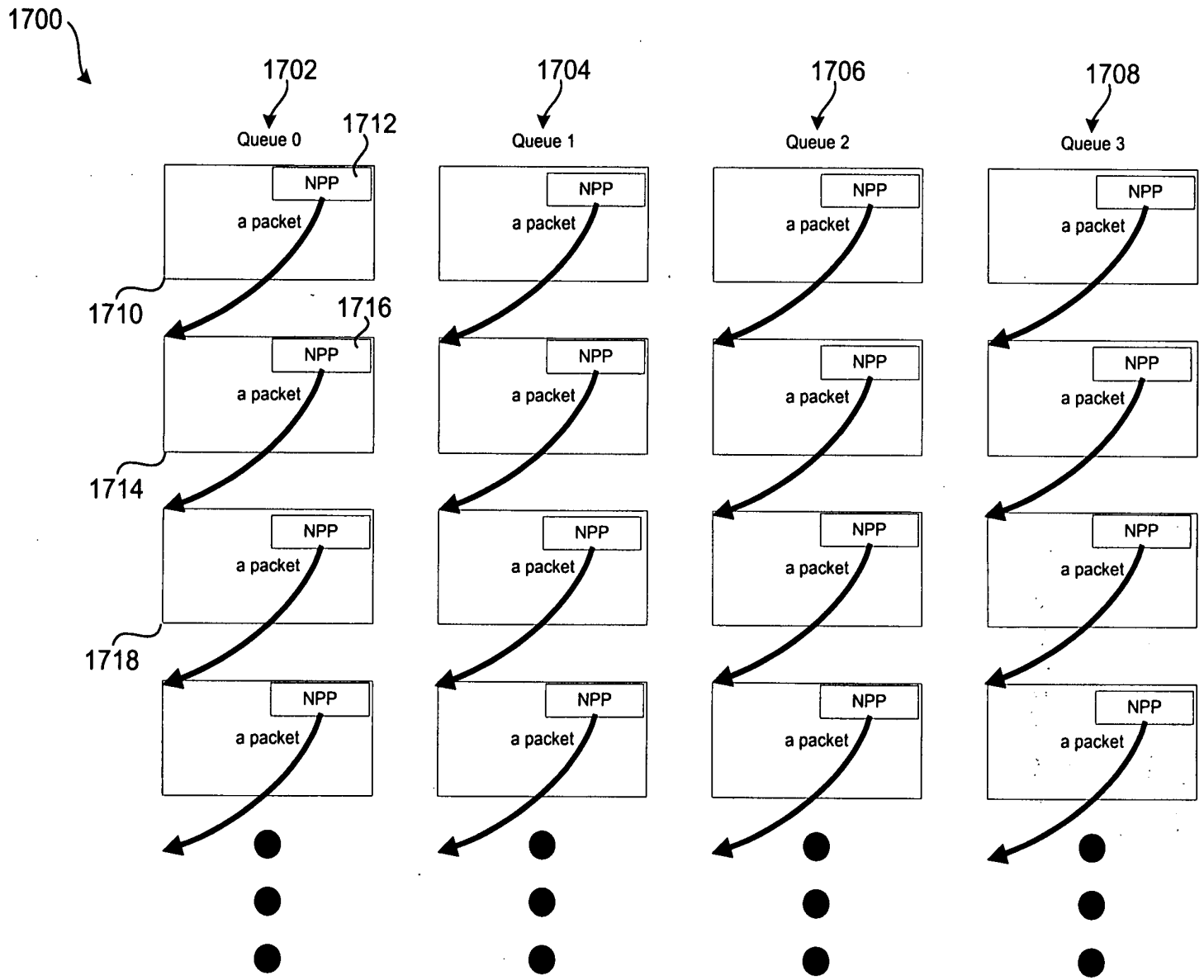

1700
1702
Queue 0
1712
1704
Queue 1
1706
Queue 2
1708
Queue 3
NPP
a packet
1710
1716
1714
1718

FIG. 17

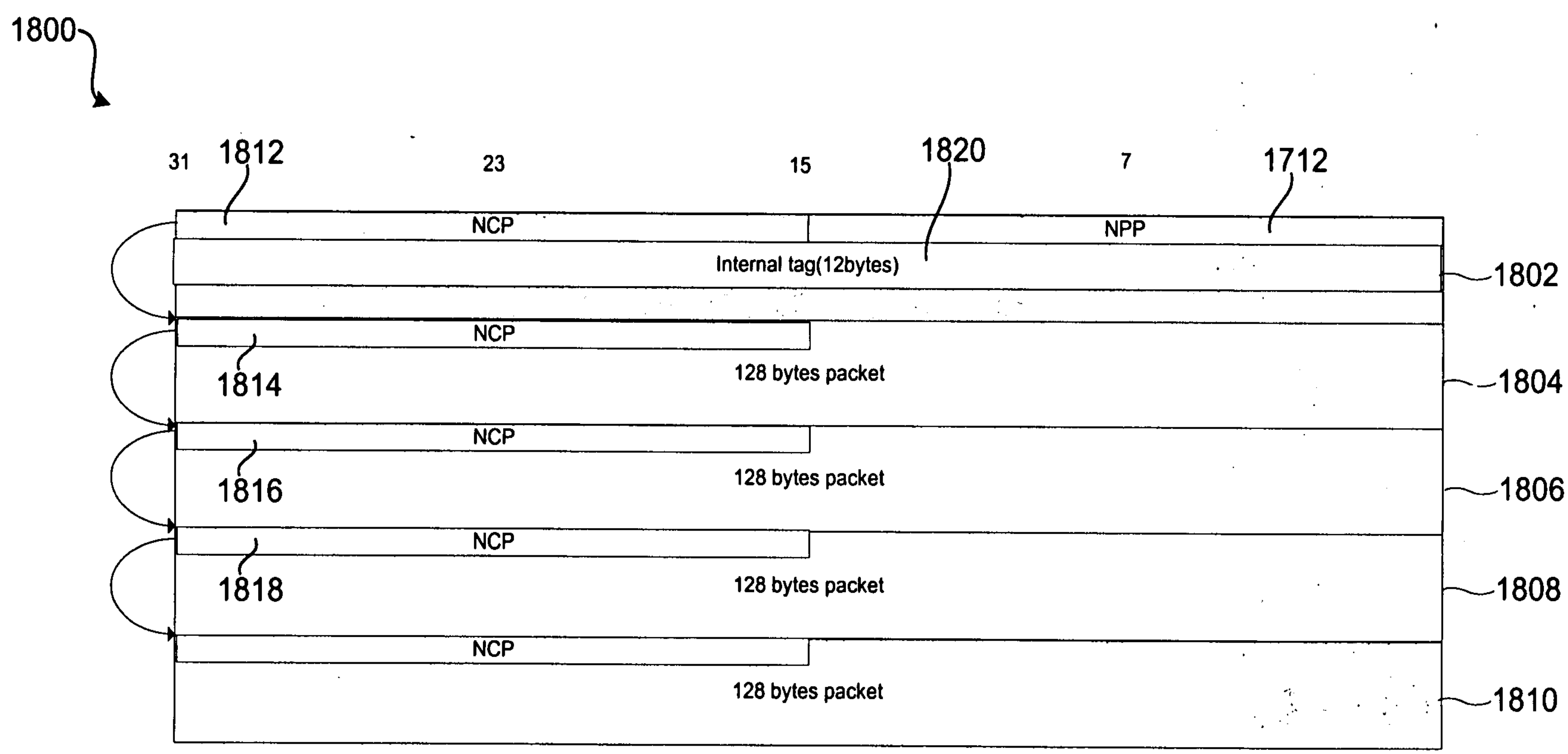


FIG. 18

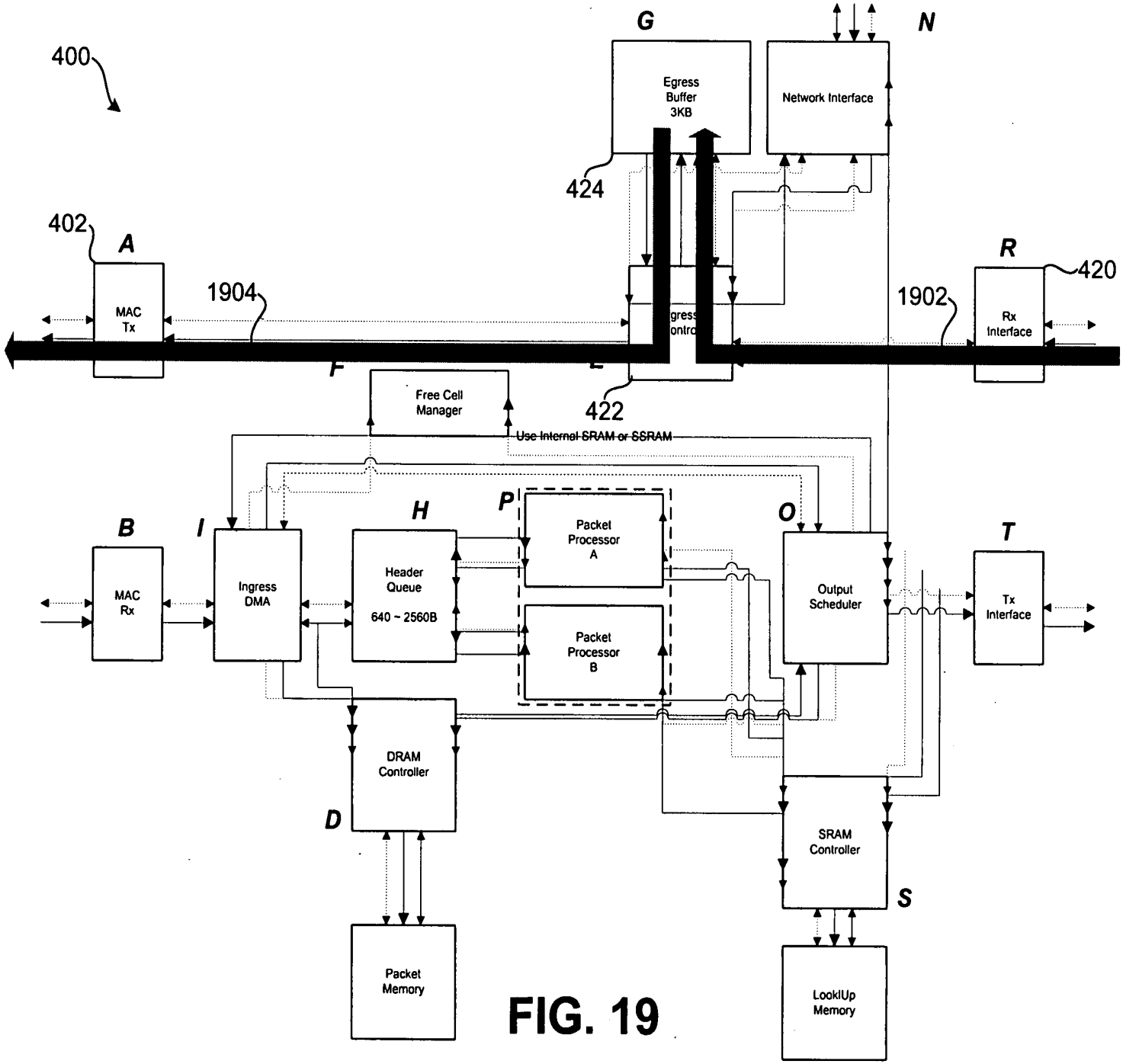


FIG. 19

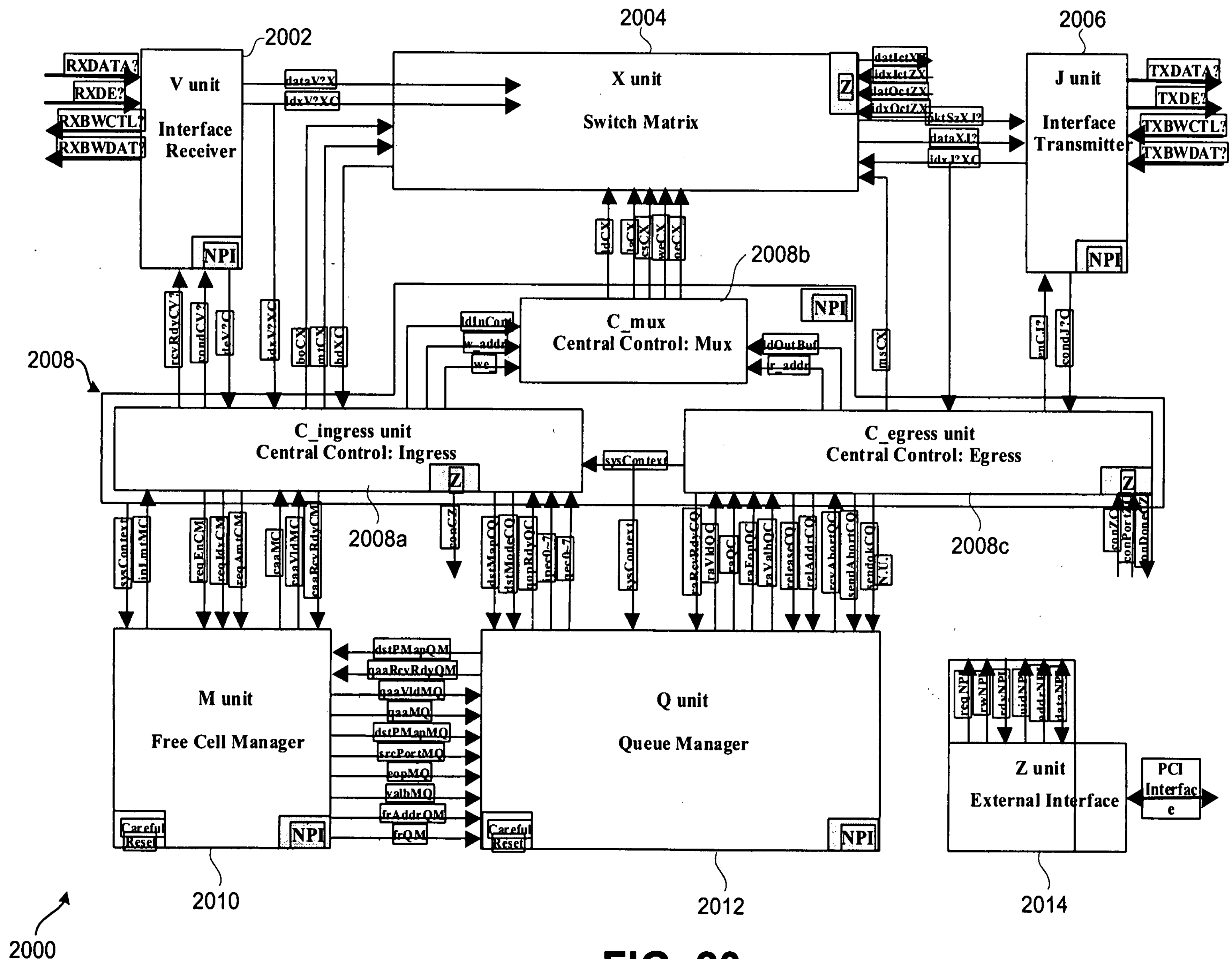

V unit
Interface Receiver
X unit
Switch Matrix
J unit
Interface Transmitter
C_mux
Central Control: Mux
C_ingress unit
Central Control: Ingress
C_egress unit
Central Control: Egress
M unit
Free Cell Manager
Q unit
Queue Manager
Z unit
External Interface
PCI Interface
NPI
Z
Careful Reset
RXDATA?
RXDE?
RXBWCTL?
RXBWDAT?
TXDATA?
TXDE?
TXBWCTL?
TXBWDAT?
dataV?X
idxV?XC
sysContext
2000
2002
2004
2006
2008
2008a
2008b
2008c
2010
2012
2014

FIG. 20

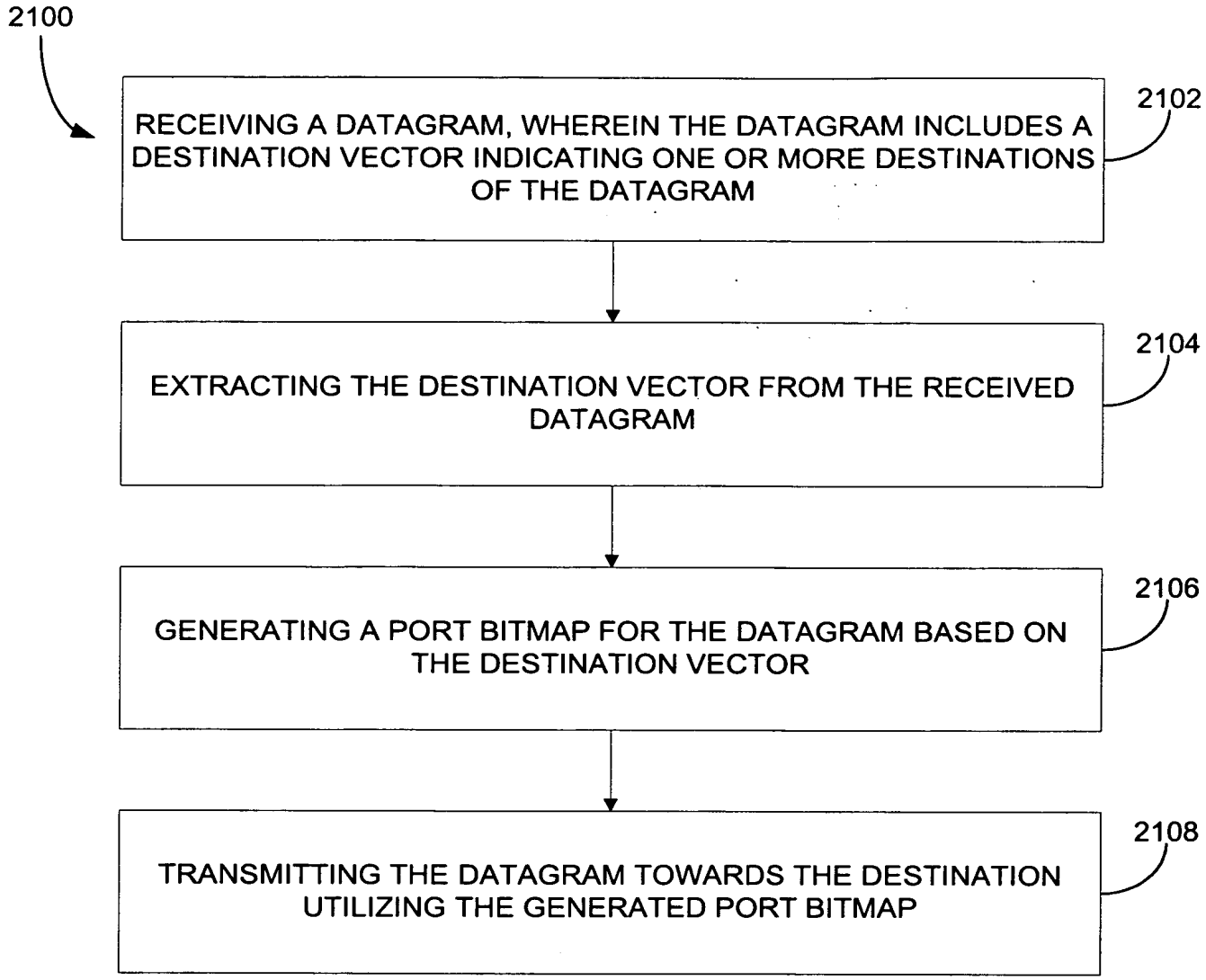

2100
RECEIVING A DATAGRAM, WHEREIN THE DATAGRAM INCLUDES A DESTINATION VECTOR INDICATING ONE OR MORE DESTINATIONS OF THE DATAGRAM
2102
EXTRACTING THE DESTINATION VECTOR FROM THE RECEIVED DATAGRAM
2104
GENERATING A PORT BITMAP FOR THE DATAGRAM BASED ON THE DESTINATION VECTOR
2106
TRANSMITTING THE DATAGRAM TOWARDS THE DESTINATION UTILIZING THE GENERATED PORT BITMAP
2108

FIG. 21

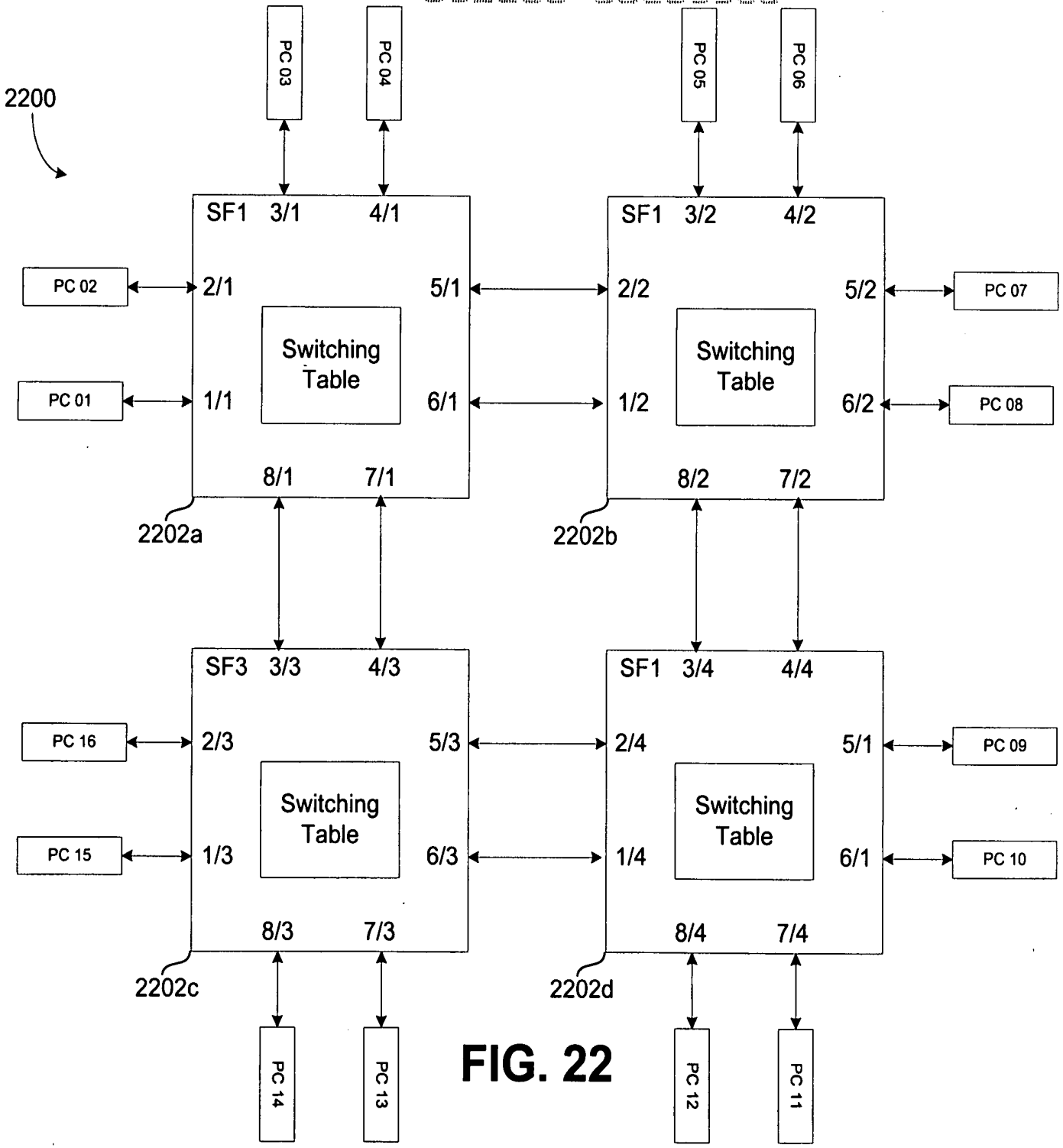

2200
PC 03
PC 04
PC 05
PC 06
SF1 3/1 4/1
SF1 3/2 4/2
PC 02
2/1
5/1
2/2
5/2
PC 07
Switching Table
Switching Table
PC 01
1/1
6/1
1/2
6/2
PC 08
8/1 7/1
8/2 7/2
2202a
2202b
SF3 3/3 4/3
SF1 3/4 4/4
PC 16
2/3
5/3
2/4
5/1
PC 09
Switching Table
Switching Table
PC 15
1/3
6/3
1/4
6/1
PC 10
8/3 7/3
8/4 7/4
2202c
2202d
PC 14
PC 13
PC 12
PC 11

FIG. 22

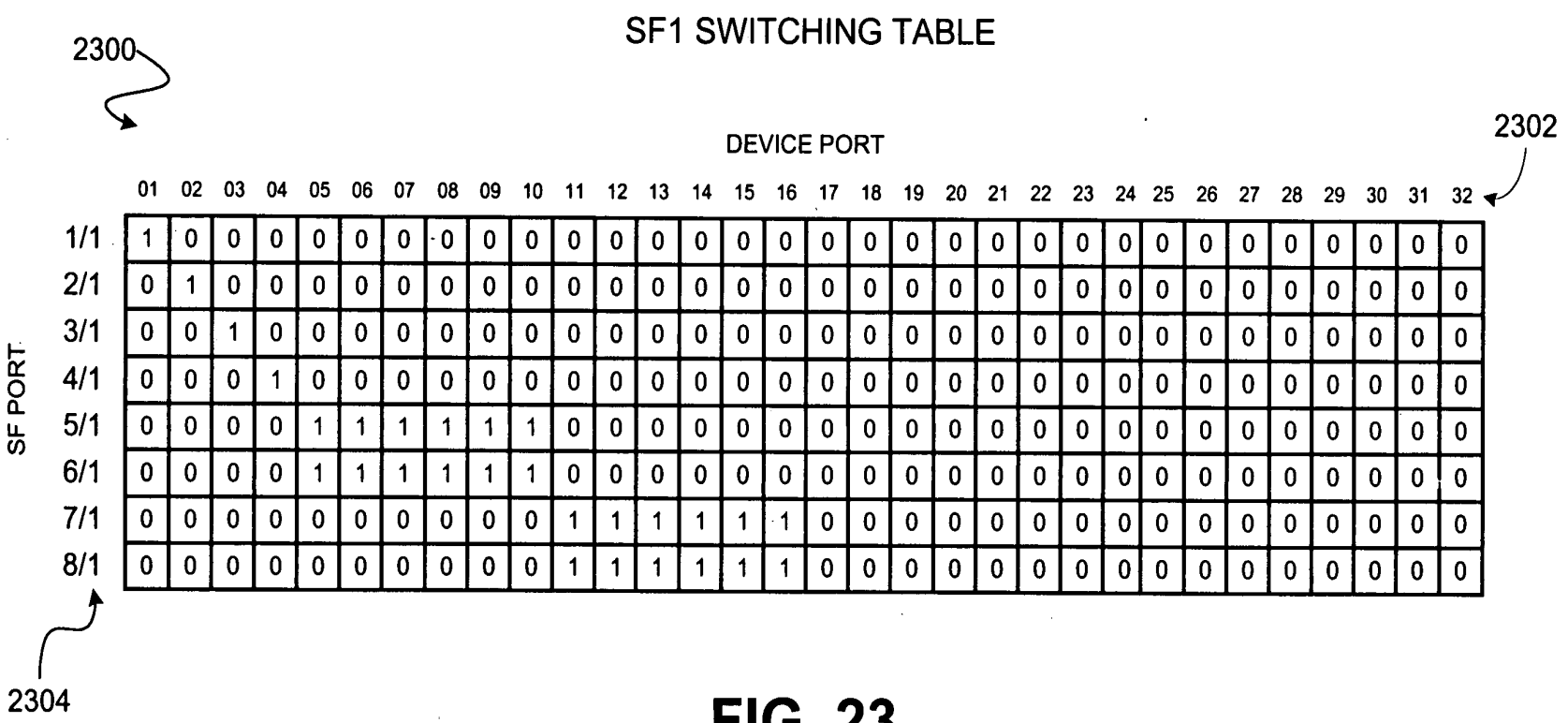


FIG. 23

SF2 SWITCHING TABLE

2400

2302

DEVICE PORT

SF PORT	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
1/2	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2/2	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3/2	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4/2	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5/2	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6/2	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7/2	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8/2	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

2304

FIG. 24

SF3 SWITCHING TABLE

2500

2302

DEVICE PORT

SF PORT	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
1/3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2/3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3/3	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4/3	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5/3	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6/3	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7/3	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8/3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

2304

FIG. 25

SF4 SWITCHING TABLE

2600

2302

DEVICE PORT

SF PORT	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
1/4	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2/4	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3/4	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4/4	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5/4	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6/4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7/4	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8/4	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

2304

FIG. 26

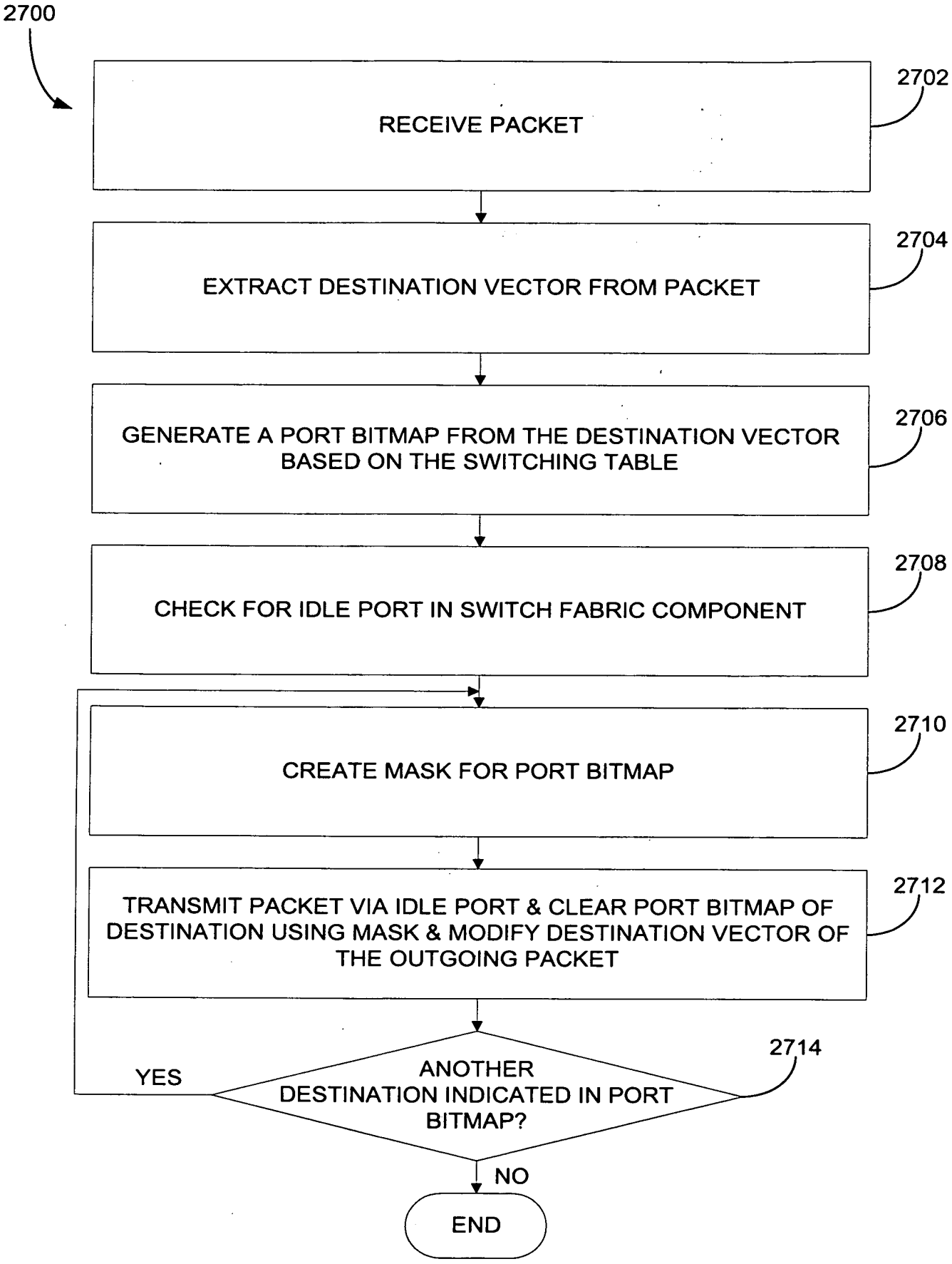

2700
RECEIVE PACKET
2702
EXTRACT DESTINATION VECTOR FROM PACKET
2704
GENERATE A PORT BITMAP FROM THE DESTINATION VECTOR BASED ON THE SWITCHING TABLE
2706
CHECK FOR IDLE PORT IN SWITCH FABRIC COMPONENT
2708
CREATE MASK FOR PORT BITMAP
2710
TRANSMIT PACKET VIA IDLE PORT & CLEAR PORT BITMAP OF DESTINATION USING MASK & MODIFY DESTINATION VECTOR OF THE OUTGOING PACKET
2712
ANOTHER DESTINATION INDICATED IN PORT BITMAP?
2714
YES
NO
END

FiG. 27

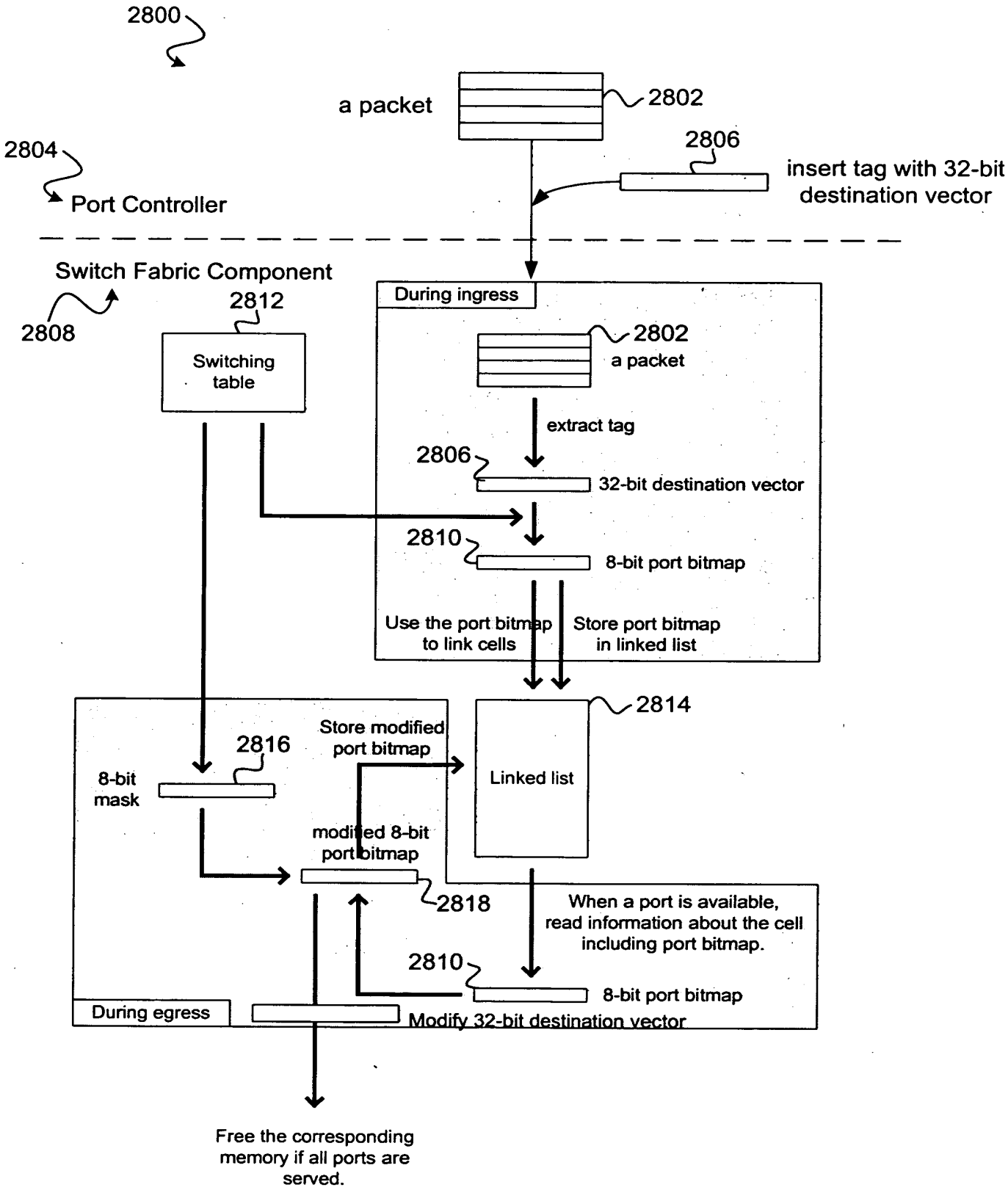

2800
a packet
2802
2804
2806
insert tag with 32-bit destination vector
Port Controller
Switch Fabric Component
2808
2812
Switching table
During ingress
2802
a packet
extract tag
2806
32-bit destination vector
2810
8-bit port bitmap
Use the port bitmap to link cells
Store port bitmap in linked list
2814
Linked list
2816
8-bit mask
Store modified port bitmap
modified 8-bit port bitmap
2818
When a port is available, read information about the cell including port bitmap.
2810
8-bit port bitmap
During egress
Modify 32-bit destination vector
Free the corresponding memory if all ports are served.

FIG. 28

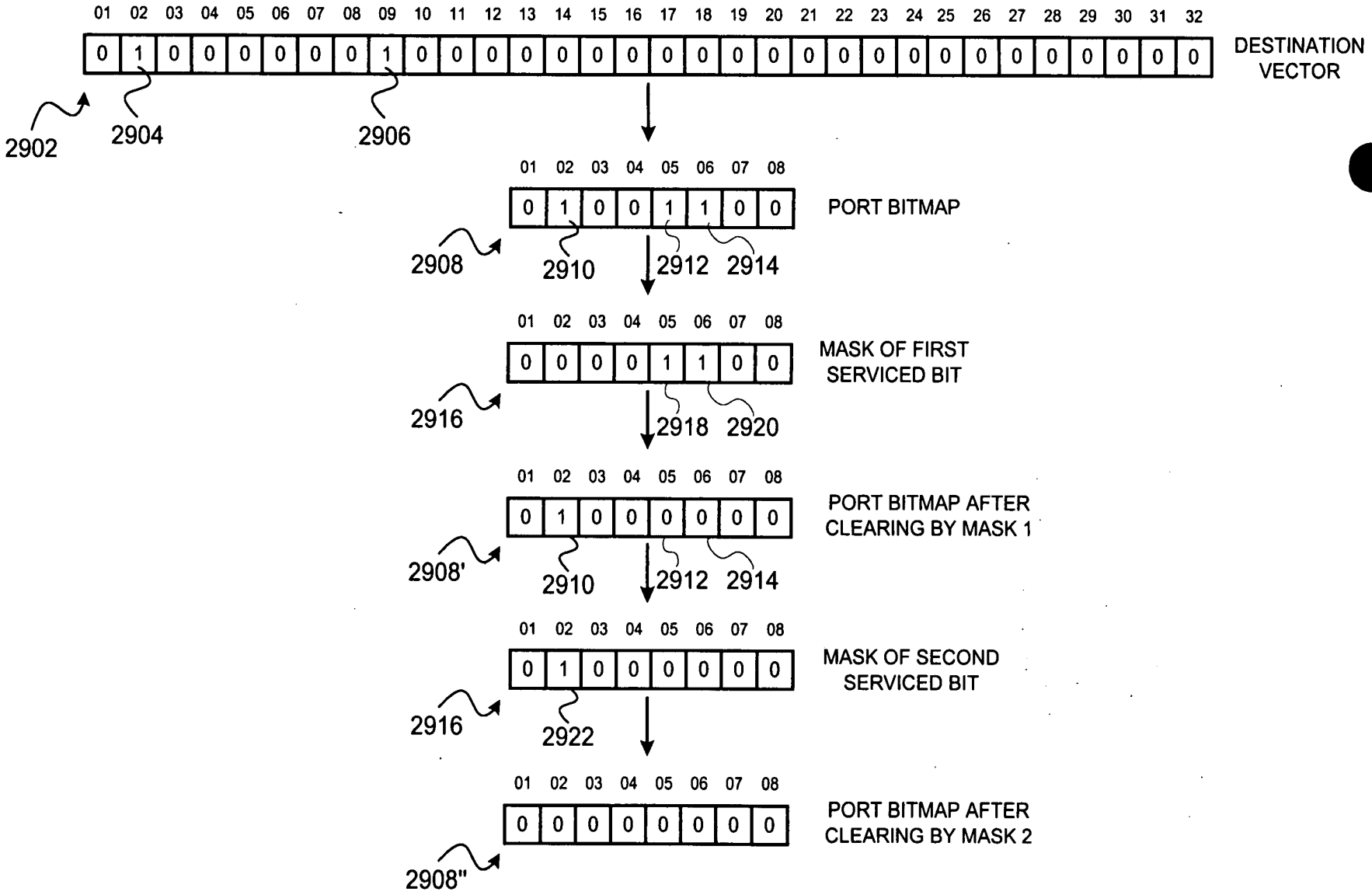

01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
DESTINATION VECTOR
2902
2904
2906
01 02 03 04 05 06 07 08
0 1 0 0 1 1 0 0
PORT BITMAP
2908
2910
2912
2914
01 02 03 04 05 06 07 08
0 0 0 0 1 1 0 0
MASK OF FIRST SERVICED BIT
2916
2918
2920
01 02 03 04 05 06 07 08
0 1 0 0 0 0 0 0
PORT BITMAP AFTER CLEARING BY MASK 1
2908'
2910
2912
2914
01 02 03 04 05 06 07 08
0 1 0 0 0 0 0 0
MASK OF SECOND SERVICED BIT
2916
2922
01 02 03 04 05 06 07 08
0 0 0 0 0 0 0 0
PORT BITMAP AFTER CLEARING BY MASK 2
2908"

FIG. 29

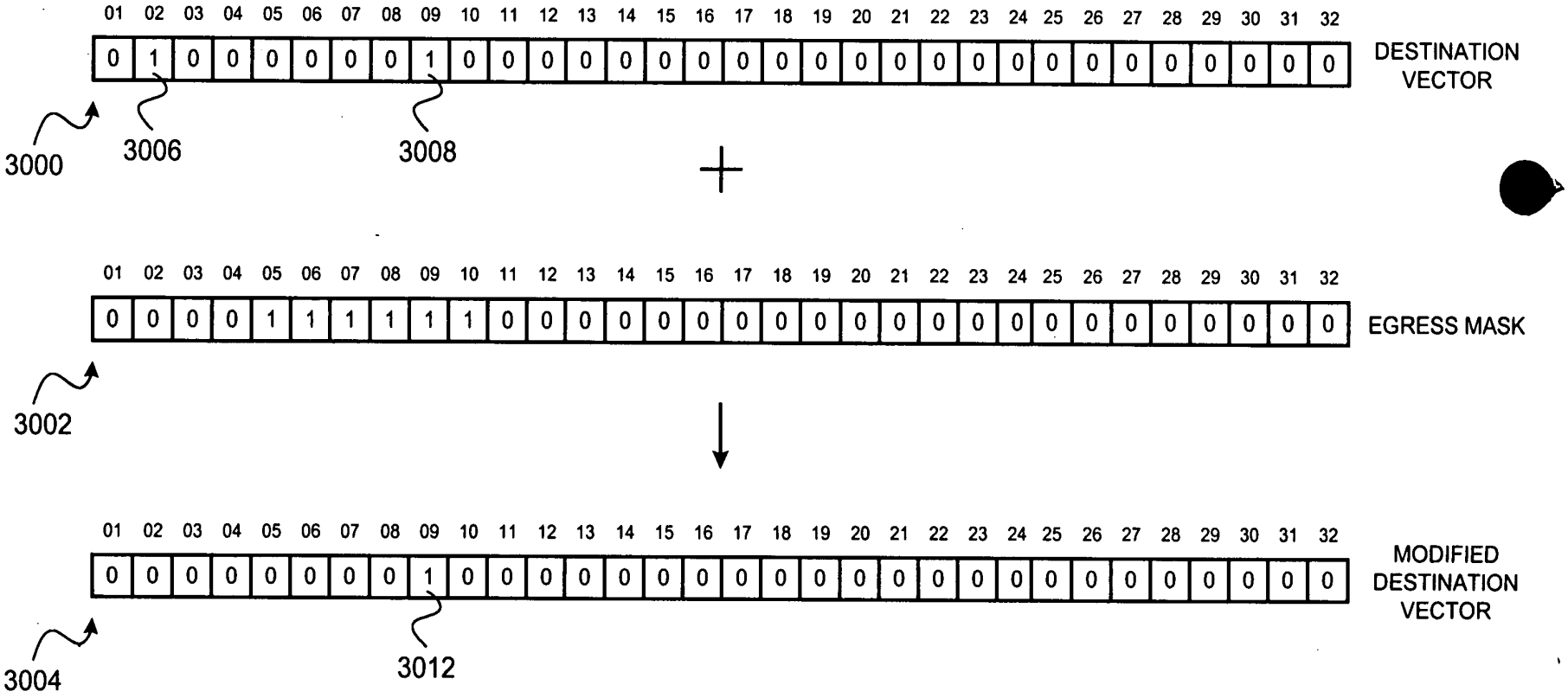

01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
DESTINATION VECTOR
3000
3006
3008
01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
0 0 0 0 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
EGRESS MASK
3002
01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
MODIFIED DESTINATION VECTOR
3004
3012

FIG. 30

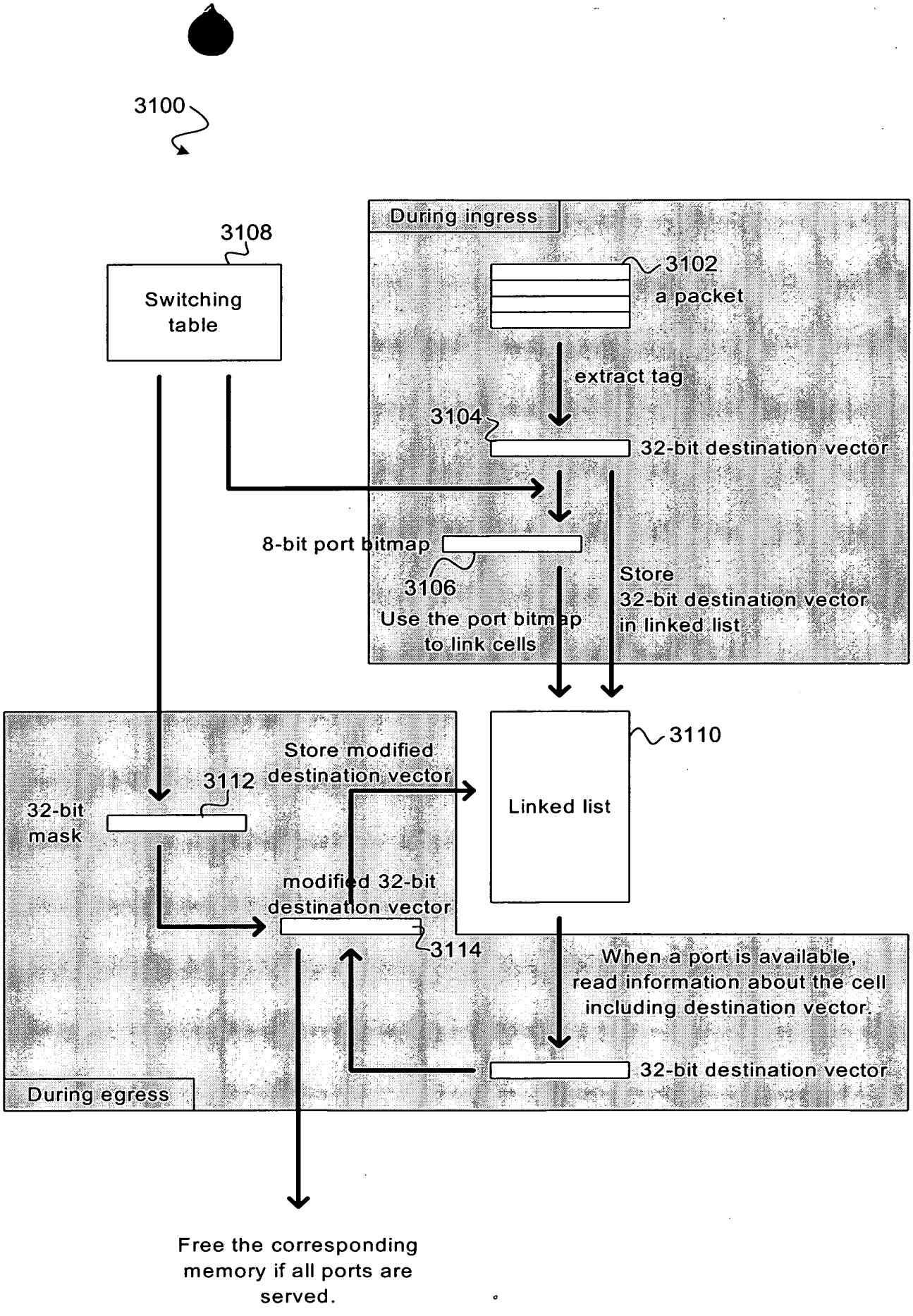

3100
3108
Switching table
During ingress
3102
a packet
extract tag
3104
32-bit destination vector
8-bit port bitmap
3106
Use the port bitmap to link cells
Store 32-bit destination vector in linked list
3110
Linked list
Store modified destination vector
3112
32-bit mask
modified 32-bit destination vector
3114
When a port is available, read information about the cell including destination vector.
32-bit destination vector
During egress
Free the corresponding memory if all ports are served.

FIG. 31

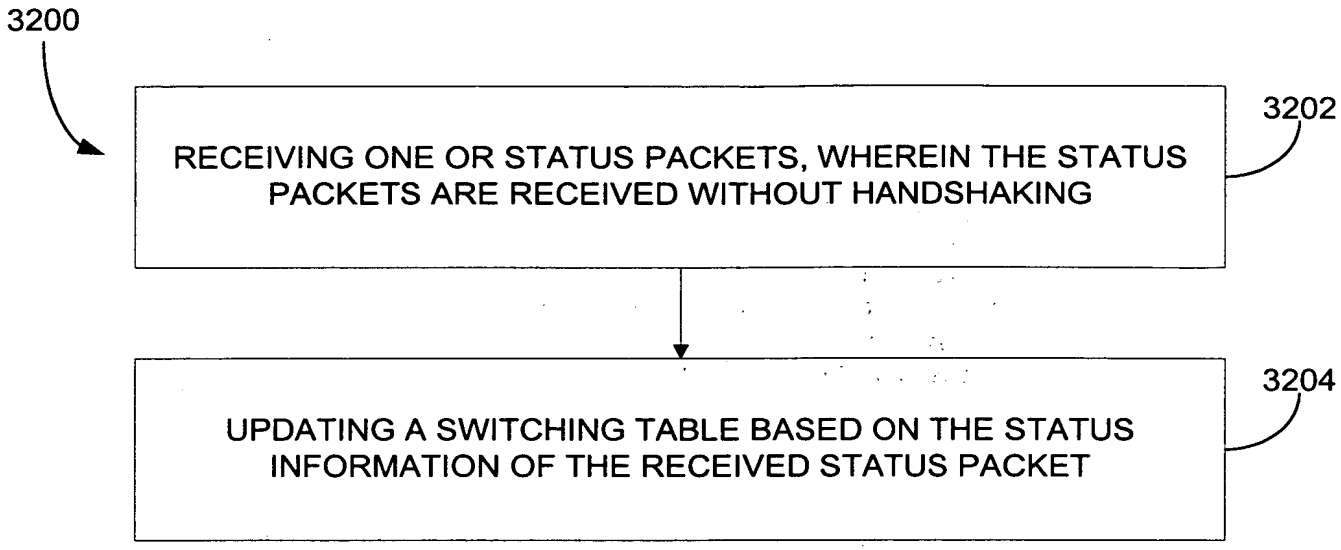

3200
RECEIVING ONE OR STATUS PACKETS, WHEREIN THE STATUS PACKETS ARE RECEIVED WITHOUT HANDSHAKING
3202
UPDATING A SWITCHING TABLE BASED ON THE STATUS INFORMATION OF THE RECEIVED STATUS PACKET
3204

FIG. 32

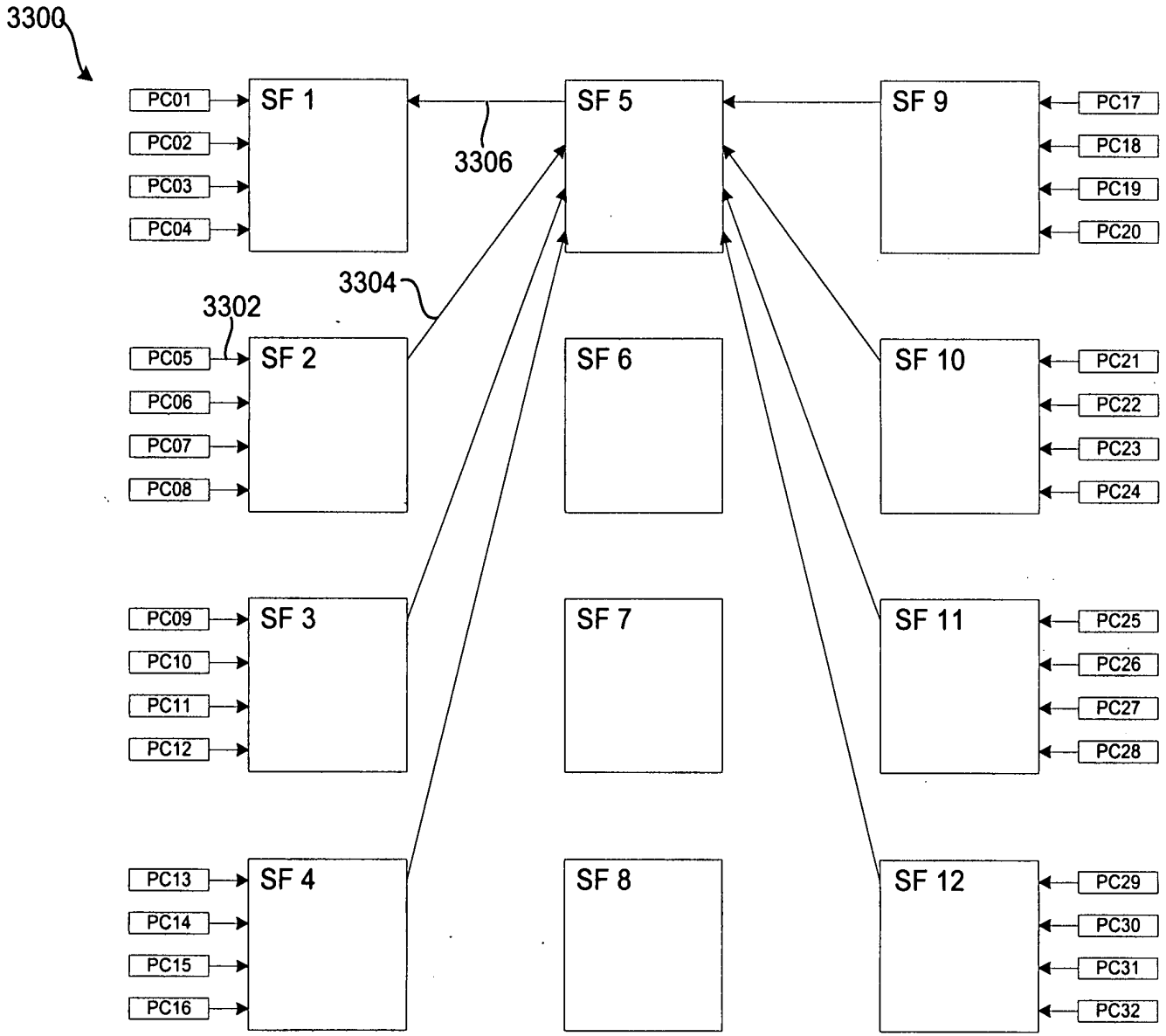


FIG. 33

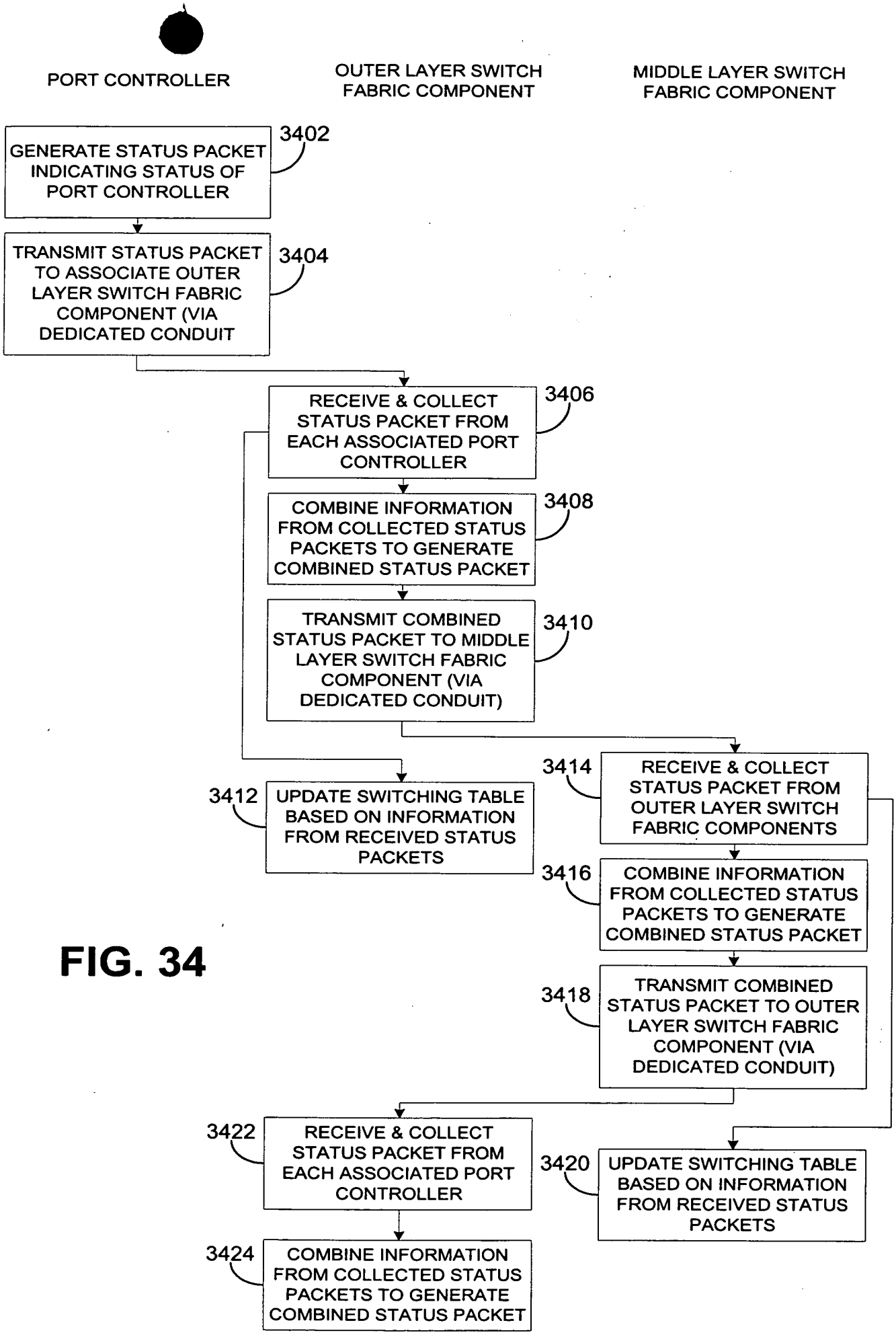

PORT CONTROLLER
OUTER LAYER SWITCH FABRIC COMPONENT
MIDDLE LAYER SWITCH FABRIC COMPONENT
GENERATE STATUS PACKET INDICATING STATUS OF PORT CONTROLLER
3402
TRANSMIT STATUS PACKET TO ASSOCIATE OUTER LAYER SWITCH FABRIC COMPONENT (VIA DEDICATED CONDUIT
3404
RECEIVE & COLLECT STATUS PACKET FROM EACH ASSOCIATED PORT CONTROLLER
3406
COMBINE INFORMATION FROM COLLECTED STATUS PACKETS TO GENERATE COMBINED STATUS PACKET
3408
TRANSMIT COMBINED STATUS PACKET TO MIDDLE LAYER SWITCH FABRIC COMPONENT (VIA DEDICATED CONDUIT)
3410
3412
UPDATE SWITCHING TABLE BASED ON INFORMATION FROM RECEIVED STATUS PACKETS
3414
RECEIVE & COLLECT STATUS PACKET FROM OUTER LAYER SWITCH FABRIC COMPONENTS
3416
COMBINE INFORMATION FROM COLLECTED STATUS PACKETS TO GENERATE COMBINED STATUS PACKET
3418
TRANSMIT COMBINED STATUS PACKET TO OUTER LAYER SWITCH FABRIC COMPONENT (VIA DEDICATED CONDUIT)
3422
RECEIVE & COLLECT STATUS PACKET FROM EACH ASSOCIATED PORT CONTROLLER
3420
UPDATE SWITCHING TABLE BASED ON INFORMATION FROM RECEIVED STATUS PACKETS
3424
COMBINE INFORMATION FROM COLLECTED STATUS PACKETS TO GENERATE COMBINED STATUS PACKET

FIG. 34

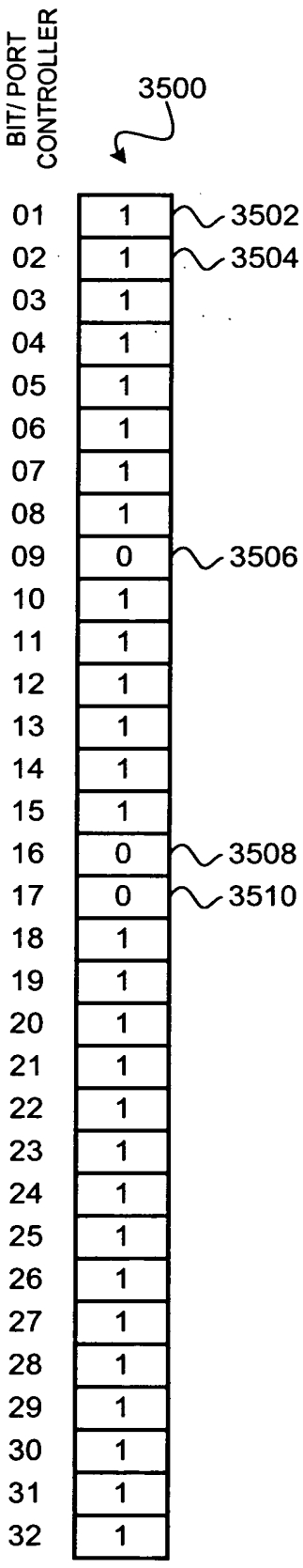

BIT/ PORT CONTROLLER
3500
01 1
02 1
03 1
04 1
05 1
06 1
07 1
08 1
09 0
10 1
11 1
12 1
13 1
14 1
15 1
16 0
17 0
18 1
19 1
20 1
21 1
22 1
23 1
24 1
25 1
26 1
27 1
28 1
29 1
30 1
31 1
32 1
3502
3504
3506
3508
3510

FIG. 35

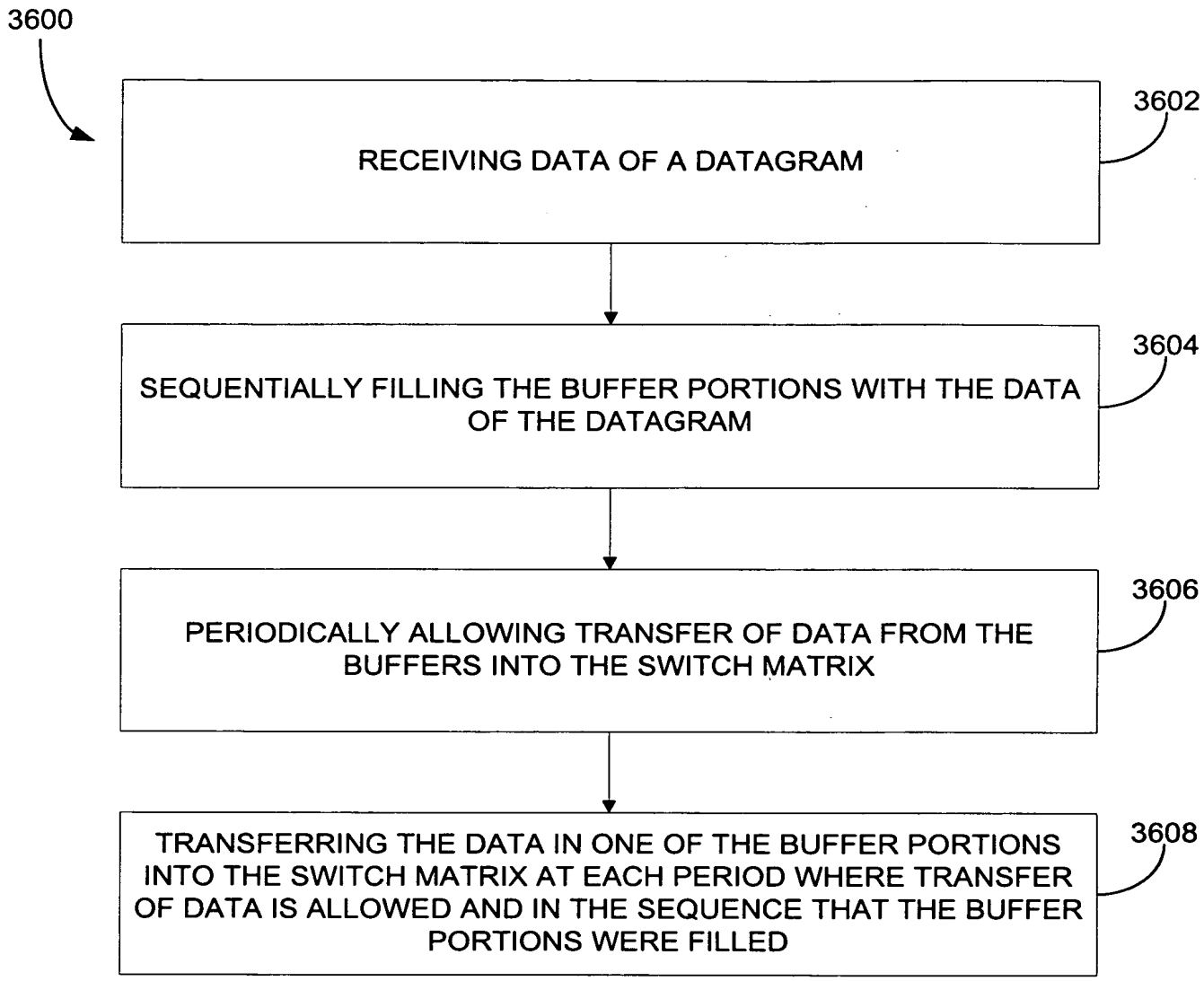

3600
RECEIVING DATA OF A DATAGRAM
3602
SEQUENTIALLY FILLING THE BUFFER PORTIONS WITH THE DATA OF THE DATAGRAM
3604
PERIODICALLY ALLOWING TRANSFER OF DATA FROM THE BUFFERS INTO THE SWITCH MATRIX
3606
TRANSFERRING THE DATA IN ONE OF THE BUFFER PORTIONS INTO THE SWITCH MATRIX AT EACH PERIOD WHERE TRANSFER OF DATA IS ALLOWED AND IN THE SEQUENCE THAT THE BUFFER PORTIONS WERE FILLED
3608

FIG. 36

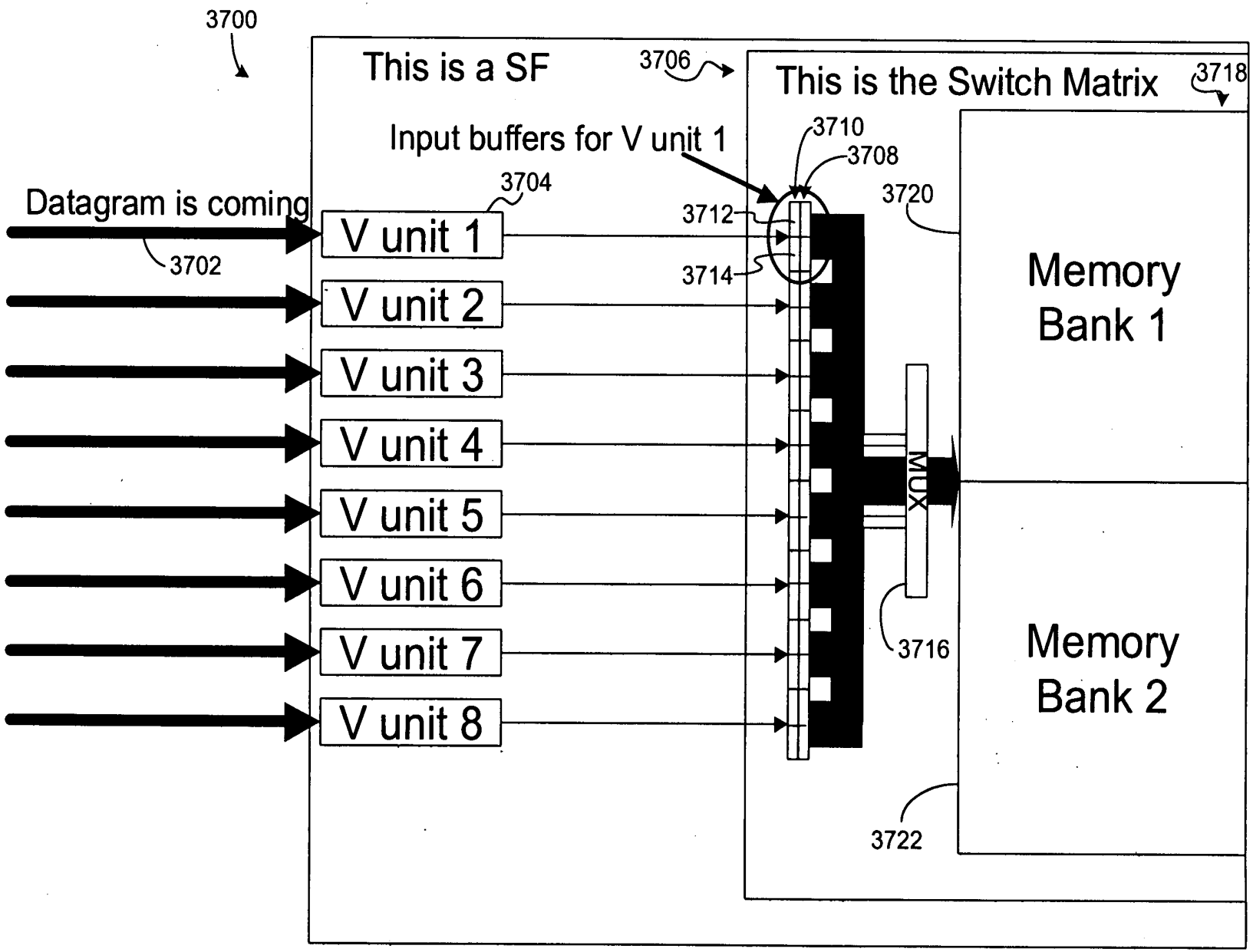

3700
This is a SF
3706
This is the Switch Matrix
3718
Input buffers for V unit 1
3710
3708
3704
Datagram is coming
3720
3712
V unit 1
3702
3714
V unit 2
V unit 3
V unit 4
V unit 5
V unit 6
V unit 7
V unit 8
MUX
Memory Bank 1
Memory Bank 2
3716
3722

FIG. 37

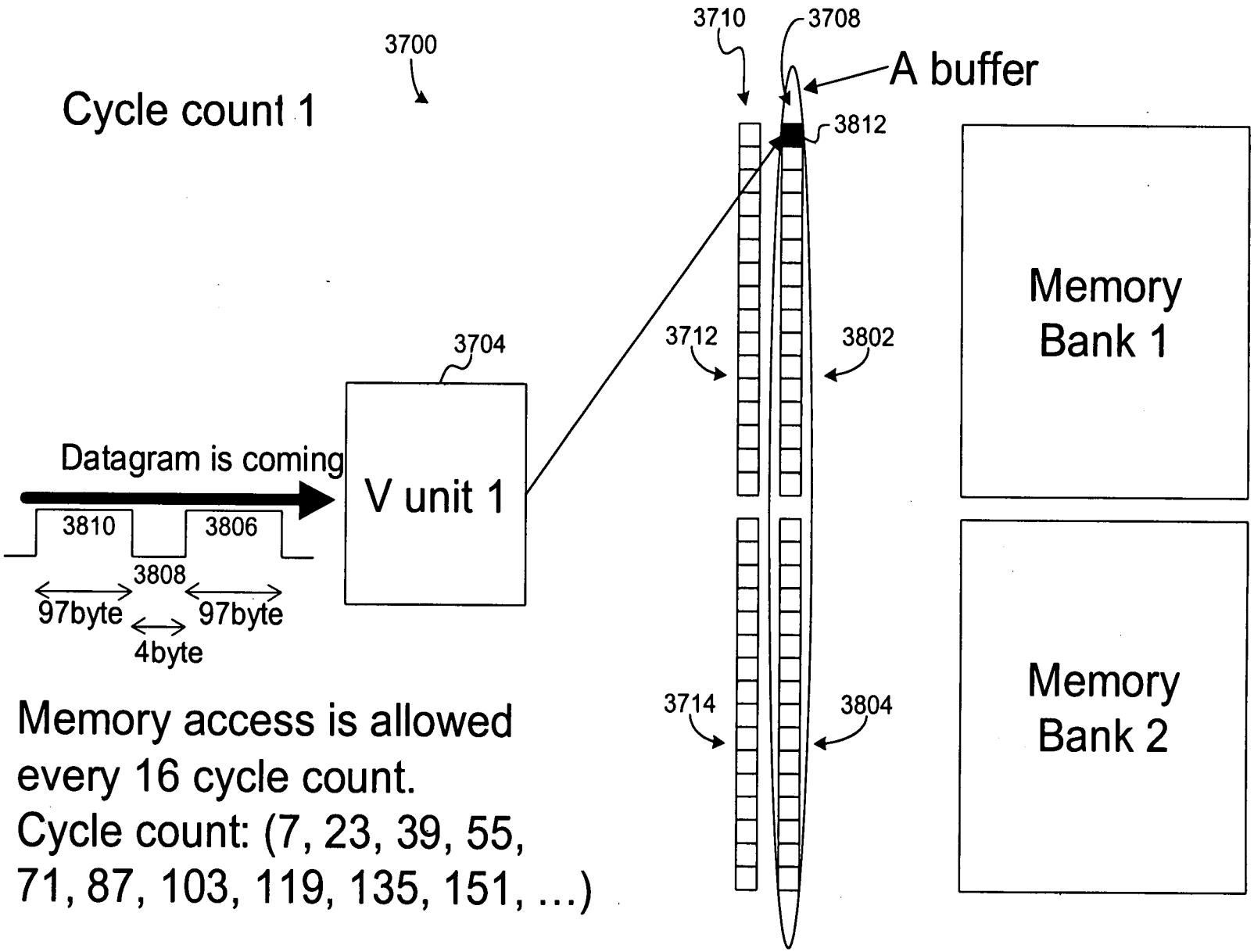

3700
Cycle count 1
3710
3708
A buffer
3812
3712
3802
3704
Datagram is coming
V unit 1
3810
3806
3808
97byte
97byte
4byte
Memory Bank 1
Memory Bank 2
3714
3804
Memory access is allowed every 16 cycle count.
Cycle count: (7, 23, 39, 55, 71, 87, 103, 119, 135, 151, ...)

FIG. 38

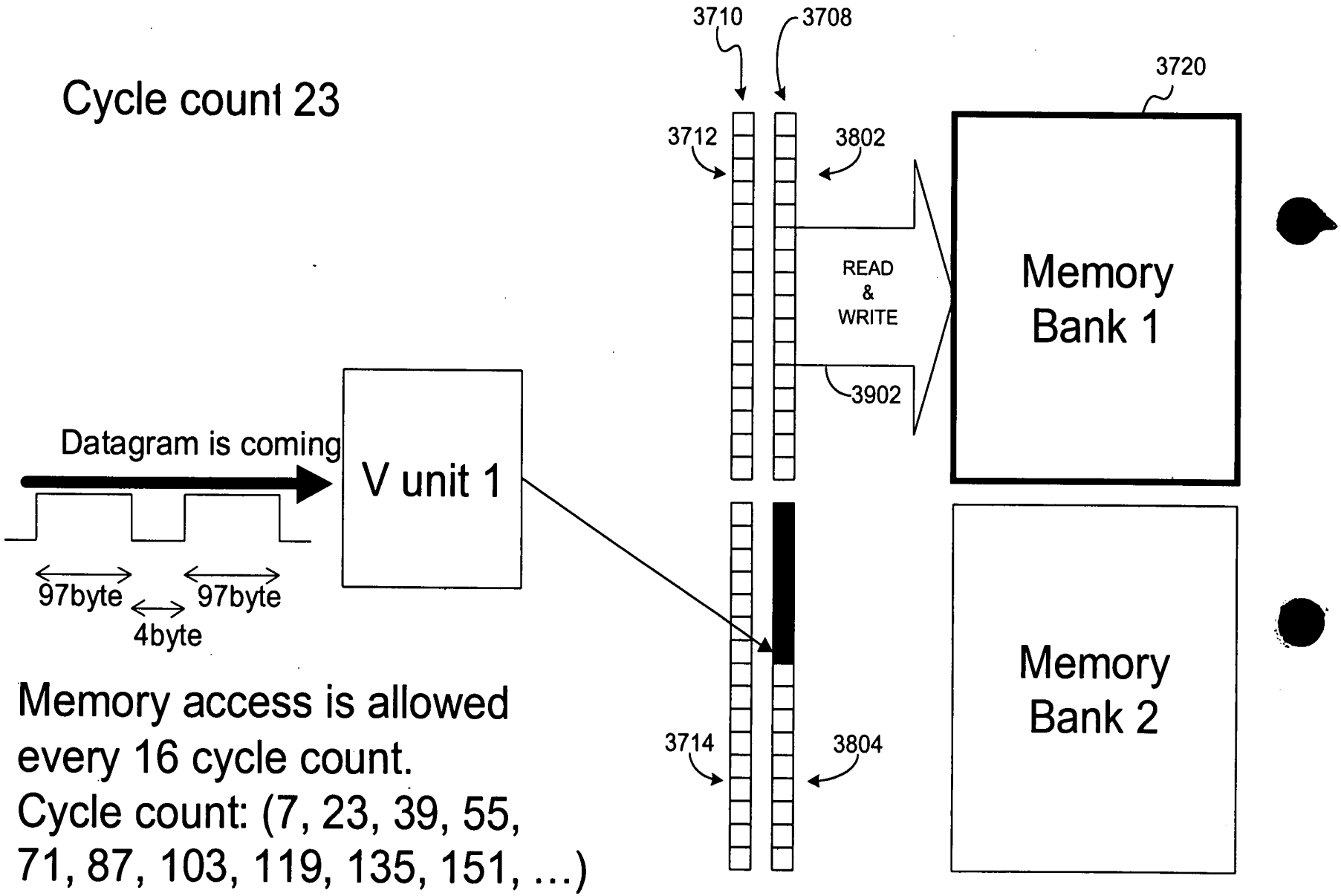

Cycle count 23
3710
3708
3720
3712
3802
READ
&
WRITE
Memory
Bank 1
3902
Datagram is coming
V unit 1
97byte
97byte
4byte
Memory access is allowed
every 16 cycle count.
Cycle count: (7, 23, 39, 55,
71, 87, 103, 119, 135, 151, ...)
3714
3804
Memory
Bank 2

FIG. 39

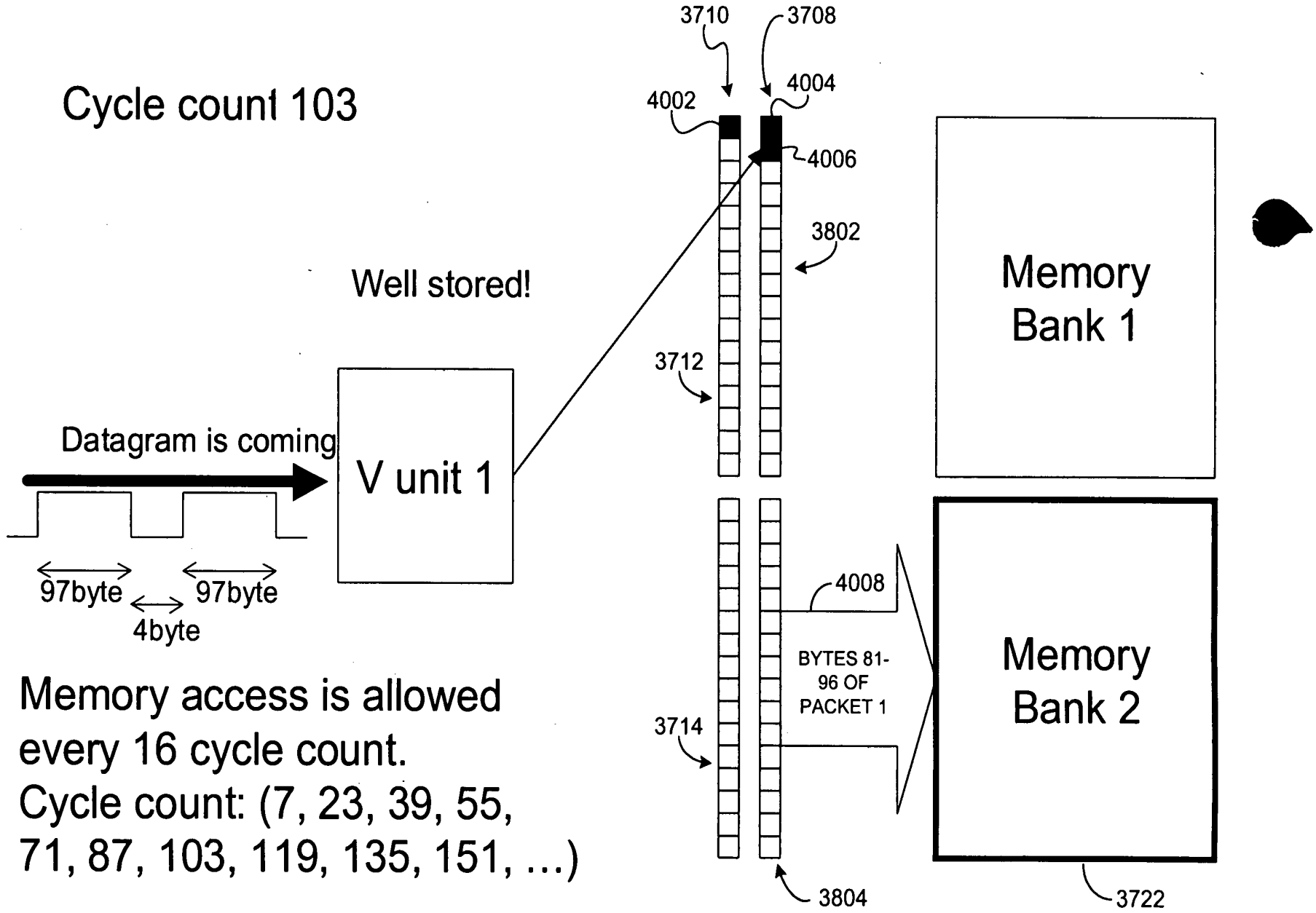

Cycle count 103
Well stored!
Datagram is coming
V unit 1
97byte
4byte
97byte
Memory access is allowed every 16 cycle count.
Cycle count: (7, 23, 39, 55, 71, 87, 103, 119, 135, 151, ...)
3710
3708
4002
4004
4006
3802
3712
Memory Bank 1
4008
BYTES 81-96 OF PACKET 1
3714
Memory Bank 2
3804
3722

FIG. 40

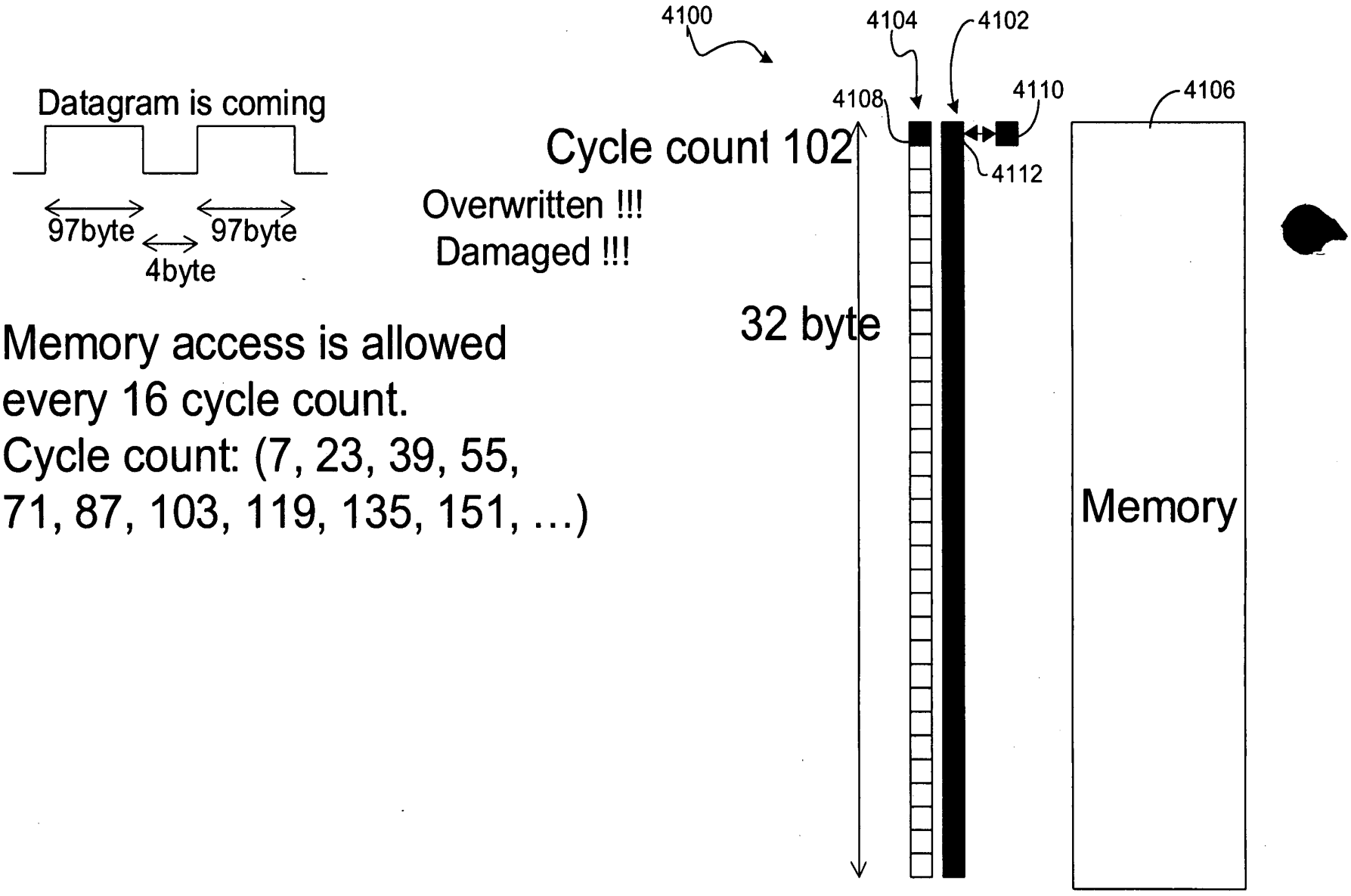
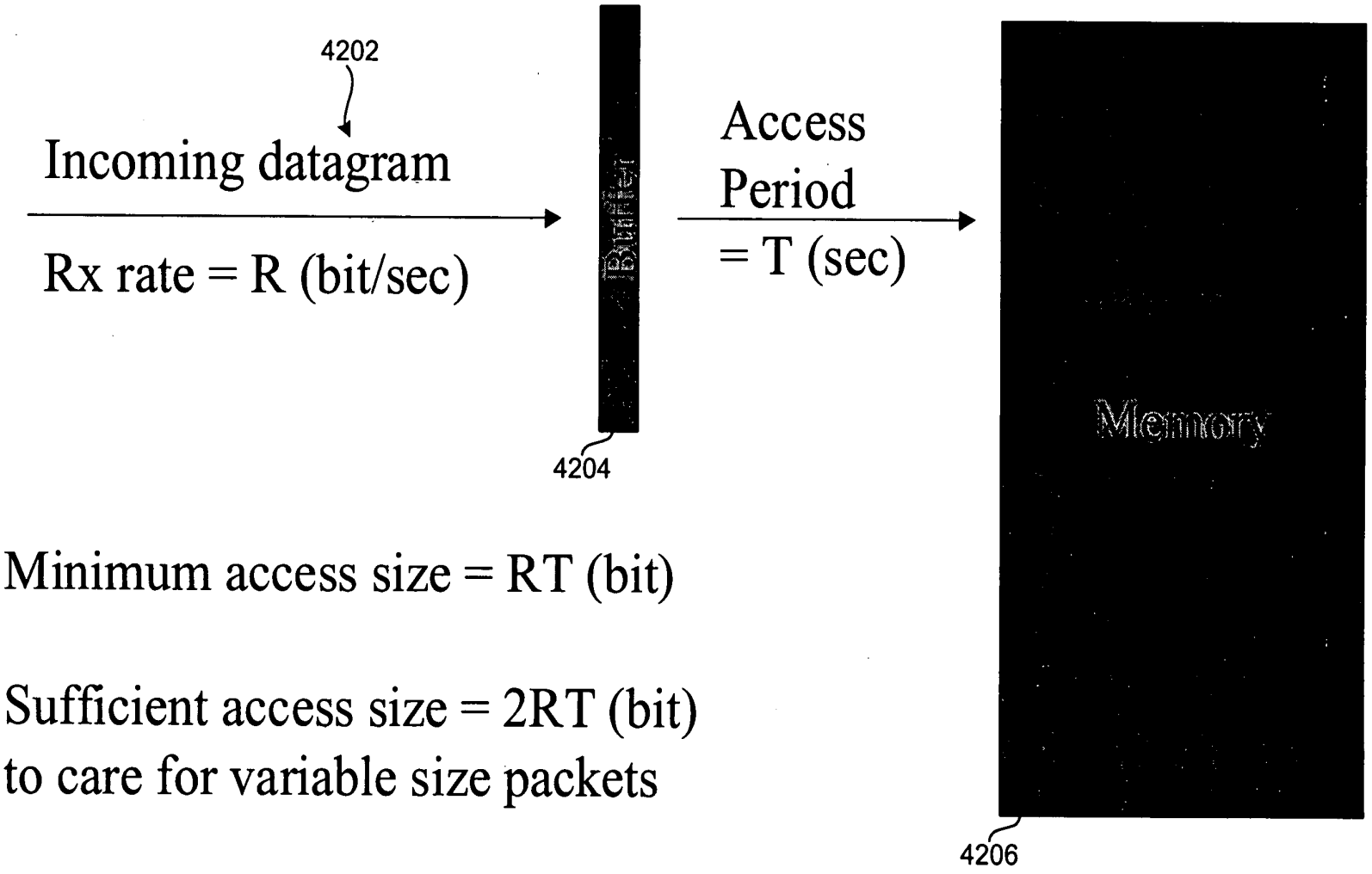

Datagram is coming
97byte
97byte
4byte
Memory access is allowed
every 16 cycle count.
Cycle count: (7, 23, 39, 55,
71, 87, 103, 119, 135, 151, ...)
4100
4104
4102
4108
4110
4106
Cycle count 102
4112
Overwritten !!!
Damaged !!!
32 byte
Memory

FIG. 41



Minimum access size = RT (bit)

Sufficient access size = 2RT (bit)
to care for variable size packets

FIG. 42

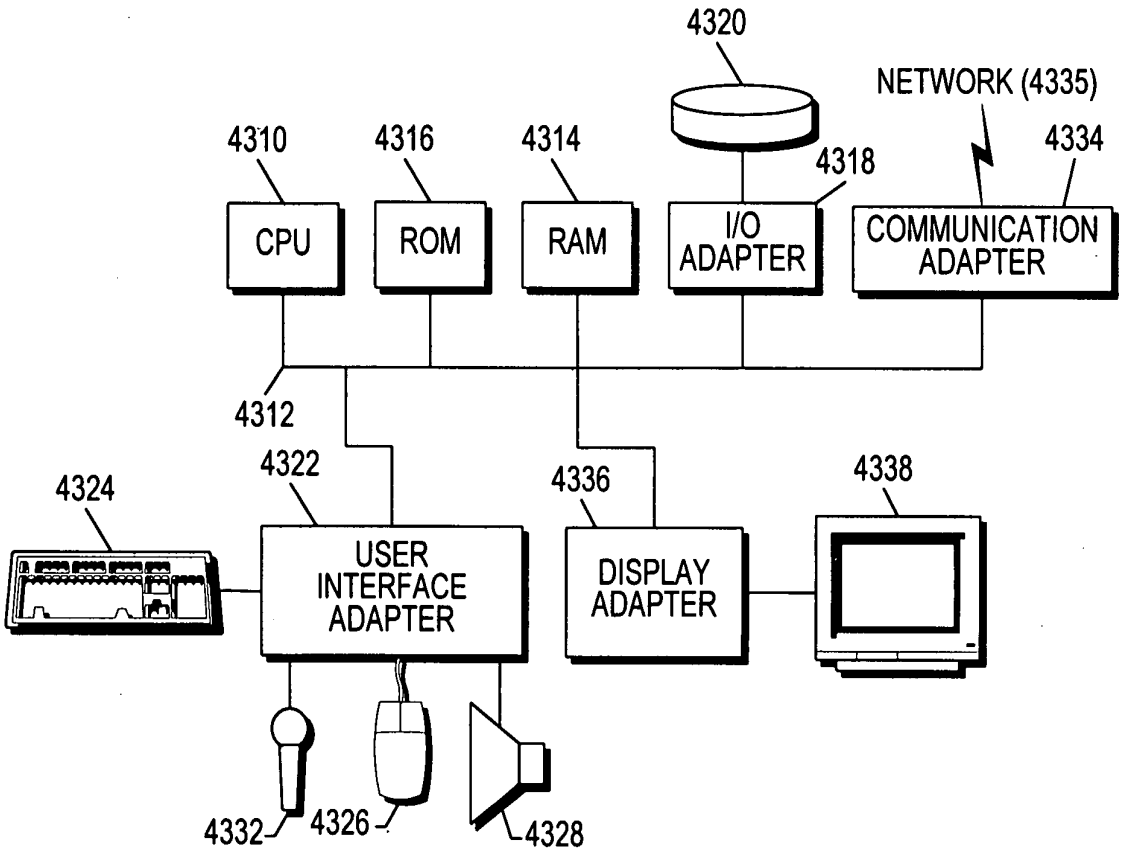


FIG. 43